Fifty-first Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
October 2 - 3, 2013

Adaptive Stochastic Convex Optimization Over Networks

Zaid J. Towfic and Ali H. Sayed

Abstract— **In this work, we study the task of distributed optimization over a network of learners in which each learner possesses a convex cost function, a set of affine equality constraints, and a set of convex inequality constraints. We propose a distributed diffusion algorithm based on penalty methods that allows the network to cooperatively optimize a global cost function, subject to all constraints and without using projection steps. We show that when sufficiently small step-sizes are employed, the expected distance between the optimal solution vector and that obtained at each node in the network can be made arbitrarily small.**

I. INTRODUCTION

Distributed convex optimization refers to the task of minimizing the aggregate sum of convex cost functions, each available at an agent of a connected network, subject to convex constraints that are also distributed across the agents. The key challenge in such problems is that each agent is only aware of its cost function and its constraints. This article proposes a fully decentralized solution that is able to minimize the aggregate cost function while satisfying all distributed constraints. The solution method is based solely on local cooperation among neighboring nodes and does not rely on the use of projection constructions.

There have been several useful studies on distributed convex optimization and estimation techniques in the literature [1]–[11]. Most existing techniques are suitable for the solution of static optimization problems, where the objective is to determine the location of a *fixed* optimal solution. These techniques become problematic in the context of adaptation and learning where the objective becomes that of solving a dynamic optimization problem in real-time from streaming data. In these dynamic scenarios, the solution methods need to track *drifts* in the location of the optimal solution that occur due to changes in the constraint conditions over time. Moreover, it is customary in several distributed optimization approaches to rely on stochastic-gradient implementations that employ decaying step-sizes that die out as time progresses [1], [3], [10], [11]. The use of decaying step-sizes is a hindrance to adaptation when it is desired to track changes in the location of the minimizer. For these reasons, we shall set all step-sizes to *constant* values in order to enable continuous adaptation and learning.

It is also customary in the literature to rely on the use of projection steps in order to ensure that the successive iterates at the nodes satisfy the convex constraints [1], [2], [9]–[11]. In some of the methods [10], [11], each node is required to know all the constraints across the entire network in order to compute the necessary projections: this requirement renders the methods non-distributed since they require the nodes to have access to global information from across the network, unless the feasible set is node-independent. The works [1], [2] develop a useful alternative method where nodes are only required to know their own constraints. However, the constraint conditions still need to be relatively simple in order for the distributed algorithm to be able to compute the necessary projections analytically (such as projecting onto the nonnegative orthant); we comment further on this situation after Eqs. (29a)–(29c).

Motivated by these considerations, in this work, we propose fully distributed solutions that employ constant step-sizes and that do not require projection steps. The solution method relies instead on the use of suitably chosen *penalty* functions and replaces the projection step by a stochastic approximation update that runs *simultaneously* with the optimization step. In the proposed technique, the nodes are only required to interact locally and there is no need for the nodes to know any of the constraints besides their own. The technique used in this work relies on the use of diffusion strategies [12], [13], which have been proven to have useful convergence and learning properties [13], [14].

Notation. Random quantities are denoted in boldface. Matrices are denoted in capital letters while vectors and scalars are denoted in small-case letters.

II. AUGMENTATION METHODS

Consider a convex optimization problem of the form:

$$\begin{aligned} \min_{w} \quad & J(w) \\ \text{subject to} \quad & g_l(w) \leq 0, \qquad l = 1, 2, \ldots, L \end{aligned} \tag{1}$$

where $w \in \mathbb{R}^M$, $\{g_1(w), \ldots, g_L(w)\}$ is a collection of convex functions, and $J(w)$ is a strongly convex function from $\mathbb{R}^M$ to $\mathbb{R}$. Augmentation incorporates the inequality constraints into the cost function and helps transform the constrained optimization problem into an unconstrained optimization problem via a *convex* barrier or penalty function $\delta(\cdot) : \mathbb{R} \to \mathbb{R}$, in the following manner:

$$\min_{w} \; J(w) + \eta \sum_{l=1}^{L} \delta(g_l(w)) \tag{2}$$

where $\eta > 0$ is a scalar parameter that controls the relative importance of adhering to the constraints. One choice for

This work was supported in part by NSF grant CCF-1011918.

The authors are with Department of Electrical Engineering, University of California, Los Angeles, CA 90095. Email: {ztowfic,sayed}@ucla.edu.

$\delta(\cdot)$ that yields an equivalent problem to (1) for any finite $\eta > 0$ is the indicator function [15, pp. 562–563]:

$$\delta^{\text{IF}}(x) = \begin{cases} 0, & x \leq 0 \\ \infty, & \text{otherwise} \end{cases} \tag{3}$$

which is convex and nondecreasing. Since the indicator function is generally nondifferentiable, approximations are used in its place. The main difference between barrier methods and penalty methods is the choice of the approximating functions.

A. Barrier Methods

Barrier methods set a "barrier" around the feasible region. One popular smooth approximation for (3) is the logarithmic barrier function:

$$\delta^{\log}(x) = \begin{cases} -\log(-x), & x < 0 \\ \infty, & \text{otherwise} \end{cases} \tag{4}$$

Solution methods that are based on (4) would require a strictly feasible initialization in order for the augmented cost in (2) to remain bounded. When the entire constraint set $\{g_1(w), \ldots, g_L(w)\}$ is not available to an agent (as happens in *distributed* constrained optimization), then it is not possible to choose a strictly feasible initializer without sharing this global information with the agents. Penalty methods avoid this difficulty.

B. Penalty Methods

In contrast to barrier methods, penalty methods give some positive penalty to solutions that fall outside the feasible set. In this case, the penalty function satisfies:

$$\delta^{\text{IP}}(x) = \begin{cases} 0, & x \leq 0 \\ > 0, & \text{otherwise} \end{cases} \tag{5}$$

One continuous, convex, and twice-differentiable choice that satisfies (5) is:

$$\delta^{\text{SIP}}(x) = \max(0, x^3) \tag{6}$$

Observe that $\delta^{\text{SIP}}(x)$ does not assume unbounded values and, therefore, penalty methods do not require a feasible solution as an initializer. While this fact implies that penalty methods are well-suited for distributed optimization scenarios, it also follows that the iterates may not remain inside the feasible region in general. We will see though that the iterates are guaranteed to converge to the feasible solution asymptotically by adding a small positive offset τ to the penalty, i.e., by considering choices of the form $\delta^{\text{IP}}(x + \tau)$.

Another advantage of penalty methods, as opposed to barrier methods, is that it is possible to easily incorporate *affine* constraints as well. Thus, consider more generally the convex optimization problem:

$$\begin{aligned} \min_{w} \quad & J(w) \\ \text{subject to} \quad & h_u(w) = 0, \qquad u = 1, 2, \ldots, U \\ & g_l(w) \leq 0, \qquad l = 1, 2, \ldots, L \end{aligned} \tag{7}$$

where the functions $h_u(w)$ are affine. This problem can also be approached as an unconstrained optimization problem using penalty functions:

$$\min_{w} \; J(w) + \eta \left[\sum_{l=1}^{L} \delta^{\text{IP}}(g_l(w) + \tau) + \sum_{u=1}^{U} \delta^{\text{EP}}(h_u(w)) \right] \tag{8}$$

where $\delta^{\text{IP}}(\cdot)$ is described in (5) while $\delta^{\text{EP}}(\cdot)$ is the convex function:

$$\delta^{\text{EP}}(x) = \begin{cases} 0, & x = 0 \\ > 0, & x \neq 0 \end{cases} \tag{9}$$

One popular choice of a continuous, convex, and twice-differentiable equality penalty function that satisfies (9) is the quadratic penalty:

$$\delta^{\text{SEP}}(x) = x^2 \tag{10}$$

Clearly, since the penalty functions are convex and the original objective function is strongly convex, the augmented cost (8) remains strongly convex. Note further that when (7) is feasible, the minimizer of (8) tends to the optimal solution of the original problem (7) as $\eta \to \infty$. This shows that it is possible to tackle both equality and inequality constraints simultaneously using penalty methods.

In the next section, we examine how penalty methods can be used in the development of distributed convex optimization algorithms for the solution of the original optimization problem (7) without the need to communicate the constraints across the agents in the network.

III. DISTRIBUTED CONSTRAINED OPTIMIZATION

Consider a connected network of agents (nodes), where each node k possesses a strongly convex cost function, $J_k(w)$, associated with it and a convex set of constraints $w \in \mathbb{W}_k$ where $w \in \mathbb{R}^M$. The objective of the network is to optimize the aggregate cost across all nodes subject to all constraints, i.e.,

$$\begin{aligned} \min_{w} \quad & J^{\text{glob}}(w) \triangleq \sum_{k=1}^{N} J_k(w) \\ \text{subject to} \quad & w \in \mathbb{W}_1, \ldots, w \in \mathbb{W}_N \end{aligned} \tag{11}$$

Each of the convex sets $\{\mathbb{W}_1, \ldots, \mathbb{W}_N\}$ is defined as the set of points w that satisfy a collection of affine equality and convex inequality constraints:

$$\mathbb{W}_k \triangleq \left\{ w : \begin{array}{ll} h_{k,u}(w) = 0, & u = 1, \ldots, U_k \\ g_{k,l}(w) \leq 0, & l = 1, \ldots, L_k \end{array} \right. \tag{12}$$

Obviously, the original optimization problem (11) can be cast as the optimization of the aggregate cost function $J^{\text{glob}}(w)$ over the common feasible set, $\mathbb{W} \triangleq \mathbb{W}_1 \cap \ldots \cap \mathbb{W}_N$:

$$\min_{w} \; J^{\text{glob}}(w) \quad \text{subject to} \quad w \in \mathbb{W} \tag{13}$$

where $\mathbb{W}$ is a convex set since the intersection of convex sets is itself convex [15, p. 36]. Assuming a solution for the above optimization problem exists (i.e, $\mathbb{W} \neq \emptyset$), we will denote it by $w^\star$. The optimal objective value is given by $J^{\text{glob}}(w^\star)$.

Remark. Although we are requiring the individual cost functions, $J_k(w)$, to be strongly convex, this condition is actually unnecessary and it is sufficient to require that at least one of the individual costs is strongly convex while all other costs can simply be convex; this condition is sufficient to ensure that the aggregate cost $J^{\text{glob}}(w)$ will remain strongly convex. Most of the results in this article will continue to hold under these weaker conditions. $\diamondsuit$

Returning to (11) and using the cost-augmentation technique described in (8), we introduce the unconstrained problem:

$$\min_{w} \; J^{\text{glob}}_{\eta,\tau}(w) \tag{14}$$

where

$$J^{\text{glob}}_{\eta,\tau}(w) \triangleq \sum_{k=1}^{N} J_k(w) + \eta \sum_{k=1}^{N} p_k(w,\tau) \tag{15}$$

and

$$p_k(w,\tau) \triangleq \sum_{l=1}^{L_k} \delta^{\text{IP}}(g_{k,l}(w)+\tau) + \sum_{u=1}^{U_k} \delta^{\text{EP}}(h_{k,u}(w)) \tag{16}$$

with $\delta^{\text{IP}}(x)$ and $\delta^{\text{EP}}(x)$ denoting convex and differentiable functions that satisfy (5) and (9), respectively. We stress that (14) is *not* an equivalent problem to (11). We will see later though that the approximation improves as $\tau \to 0$ and $\eta \to \infty$. Since $J^{\text{glob}}(w)$ is strongly convex, the cost (14) will also be strongly-convex and will have a unique optimizer for each pair $(\eta, \tau) \succeq 0$. We shall denote this optimal solution to (14) by $w^o(\eta,\tau)$, which is parameterized in terms of (η,τ). Our task is now two-fold: (1) to motivate a fully distributed algorithm to solve (14) and determine $w^o(\eta,\tau)$ by using diffusion strategies, and (2) to characterize the distance between $w^o(\eta,\tau)$ and the desired optimizer $w^\star$ of (11). We will establish that by choosing the algorithm's parameters appropriately, it is possible to obtain an arbitrarily accurate approximation for $w^\star$.

A. Diffusion Strategy for Distributed Optimization

Thus, consider the optimization problem given by (14). Its aggregate cost can be expressed as the sum of local cost functions as follows:

$$J^{\text{glob}}_{\eta,\tau}(w) \triangleq \sum_{k=1}^{N} J'_{k,\eta,\tau}(w) \tag{17}$$

where

$$J'_{k,\eta,\tau}(w) \triangleq J_k(w) + \eta \cdot p_k(w,\tau) \tag{18}$$

and $p_k(w,\tau)$ is defined in (16). Observe that each function $J'_{k,\eta,\tau}(w)$ depends only on agent k's information: cost function $J_k(w)$ and constraint set $\mathbb{W}_k$. This situation falls within the framework of unconstrained diffusion optimization developed in [12], [13], which extends the original diffusion formulations of [16], [17]. Following similar arguments to those employed in these references, we conclude that one way to seek the minimizer of (17) is for each node to run the following diffusion strategy:

$$\psi_{k,i} = w_{k,i-1} - \mu \cdot \nabla_w J'_{k,\eta,\tau}(w_{k,i-1}) \tag{19a}$$

$$w_{k,i} = \sum_{\ell=1}^{N} a_{\ell k} \psi_{\ell,i} \tag{19b}$$

In (19a)-(19b), the vector $w_{k,i-1}$ denotes the estimate for $w^o(\eta,\tau)$ at node k at iteration $i-1$. This iterate is first updated via the gradient-descent update (19a) with step-size $\mu > 0$ to the intermediate value $\psi_{k,i}$. All other nodes in the network perform a similar update simultaneously by using their gradient vectors. Subsequently, each node k uses (19b) to combine, in a convex manner, the intermediate estimates from its neighbors. This step results in the updated estimate $w_{k,i}$ and the process repeats itself. The nonnegative coefficients $\{a_{\ell k}\}$ are chosen to satisfy the conditions:

$$a_{\ell k} = 0, \quad \text{when } \ell \text{ and } k \text{ are not neighbors} \tag{20a}$$

$$\sum_{\ell=1}^{N} a_{\ell k} = 1, \quad k = 1, \ldots, N \tag{20b}$$

If we collect these coefficients into a matrix $A = [a_{\ell k}]$, then condition (20b) implies that A is left-stochastic (i.e., it satisfies $A^{\mathsf{T}} \mathbb{1}_N = \mathbb{1}_N$, where $\mathbb{1}_N$ is the vector with all entries equal to one).

Evaluating the gradient vector from (18) and substituting into (19a) we get:

$$\psi_{k,i} = w_{k,i-1} - \mu \cdot \nabla_w J_k(w_{k,i-1}) - \mu\eta \cdot \nabla_w p_k(w,\tau) \tag{21}$$

for differentiable penalty functions. Expression (21) indicates that the update from $w_{k,i-1}$ to $\psi_{k,i}$ involves two gradients: the original gradient vector, $\nabla_w J_k(\cdot)$, and the gradient vector of the penalty function. We can incorporate these update terms into $w_{k,i-1}$ in various orders. One convenient way is to split the update into two parts: first we move from $w_{k,i-1}$ to an intermediate vector $\zeta_{k,i}$ in the opposite direction of the gradient vector of $J_k(\cdot)$. Subsequently, we incorporate the correction by the penalty gradient to obtain $\psi_{k,i}$, i.e.,

$$\zeta_{k,i} = w_{k,i-1} - \mu \cdot \nabla_w J_k(w_{k,i-1}) \tag{22a}$$

$$\psi_{k,i} = \zeta_{k,i} - \mu\eta \cdot \nabla_w p_k(w_{k,i-1},\tau) \tag{22b}$$

It is generally expected that the intermediate iterate $\zeta_{k,i}$ generated by (22a) is a better estimate for $w^o(\eta,\tau)$ than $w_{k,i-1}$. This motivates us to replace $w_{k,i-1}$ in the rightmost term in (22b) by $\zeta_{k,i}$ to get:

$$\zeta_{k,i} = w_{k,i-1} - \mu \cdot \nabla_w J_k(w_{k,i-1}) \tag{23}$$

$$\psi_{k,i} = \zeta_{k,i} - \mu\eta \cdot \nabla_w p_k(\zeta_{k,i},\tau) \tag{24}$$

This substitution is reminiscent of incremental-type arguments in gradient descent algorithms [18].

Now, combining (23)–(24) with (19b), we arrive at what we shall refer to as the *penalized* Adapt-then-Combine (ATC) algorithm:

$$\zeta_{k,i} = w_{k,i-1} - \mu \cdot \nabla_w J_k(w_{k,i-1}) \tag{25a}$$

$$\psi_{k,i} = \zeta_{k,i} - \mu\eta \cdot \nabla_w p_k(\zeta_{k,i}, \tau) \tag{25b}$$

$$w_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \psi_{\ell,i} \tag{25c}$$

where $\mathcal{N}_k$ denotes the neighborhood of node k. It is also possible to interchange the order in which steps (19a)–(19b) are performed, with combination performed prior to adaptation. Following similar arguments to the above, we can motivate the following alternative *penalized* Combine-and-Adapt (CTA) algorithm:

$$\psi_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{\ell,i-1} \tag{26a}$$

$$\zeta_{k,i} = \psi_{k,i-1} - \mu \cdot \nabla_w J_k(\psi_{k,i-1}) \tag{26b}$$

$$w_{k,i} = \zeta_{k,i} - \mu\eta \cdot \nabla_w p_k(\zeta_{k,i}, \tau) \tag{26c}$$

Observe that in both penalized ATC and CTA algorithms, there is an explicit step to move along the gradient of the penalty function. This step can be thought of as performing a single incremental "projection" step along agent k's constraints [19, pp. 20-21].

B. Comparison with the Consensus Construction

It is instructive to compare the penalized CTA algorithm (26a)–(26c) with the consensus-based algorithm used, for example, in [10] to solve similar constrained optimization problems. The algorithm is reproduced below using our notation:

$$\psi_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{\ell,i-1} \tag{27a}$$

$$\zeta_{k,i} = \psi_{k,i-1} - \mu \cdot \nabla_w J_k(w_{k,i-1}) \tag{27b}$$

$$w_{k,i} = P_{\mathbb{W}_1 \cap \ldots \cap \mathbb{W}_N}[\zeta_{k,i}] \tag{27c}$$

where the notation $P_{\mathbb{X}}[y]$ denotes the operation of projecting the vector y onto the set $\mathbb{X}$:

$$P_{\mathbb{X}}[y] \triangleq \arg\min_{x \in \mathbb{X}} \|x - y\| \tag{28}$$

Observe that the gradient vector in (27b) is evaluated at the old iterate, $w_{k,i-1}$, and not at the updated iterate $\psi_{k,i-1}$ as in (26b). Moreover, the projection step (27c) assumes global knowledge of the full feasible set $\mathbb{W}$ by node k, which runs counter to the objective of a fully distributed solution. In addition, unless the constraints are simple, the actual projection in (27c) is usually found via augmentation methods such as the barrier method discussed in Sec. II-A, and enough iterations need to be executed offline.

C. Comparison with Projection-Based Constructions

Another distributed algorithm is developed in [1]; it relies on a diffusion structure similar to the penalized CTA form albeit with two important differences: step (26c) is replaced by the local projection step (29c) shown below and the constant step-size in step (26b) is replaced by an iteration-dependent step-size in step (29b):

$$\psi_{k,i-1} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{\ell,i-1} \tag{29a}$$

$$\zeta_{k,i} = \psi_{k,i-1} - \mu(i) \cdot \nabla_w J_k(\psi_{k,i-1}) \tag{29b}$$

$$w_{k,i} = P_{\mathbb{W}_k}[\zeta_{k,i}] \tag{29c}$$

In this solution, each node would project only onto its constraint set $\mathbb{W}_k$, as indicated by (29c). However, the constraint set $\mathbb{W}_k$ is required to consist of simple constraints whose projections (29c) can be computed analytically, such as the projection onto the non-negative orthant. Moreover, note that step (29b) utilizes a diminishing step-size, which limits the adaptation ability of the network in tracking drifting constraints. In contrast, the diffusion strategy (26a)–(26c) employs a constant step-size. When this is done, the dynamics of the distributed algorithm is changed in a nontrivial manner because the gradient update term will not die out anymore with time as happens with decaying step-size implementations.

IV. MAIN ASSUMPTIONS

In this section, we examine the performance of the penalized ATC and CTA diffusion strategies (25a)-(25c) and (26a)-(26c). We do not limit our analysis to deterministic optimization problems, but consider more general stochastic gradient approximation problems where the true gradient vectors, $\nabla_w J_k(\cdot)$, are replaced by approximations, say, $\widehat{\nabla_w} J_k(\cdot)$.

We model the approximate gradient direction as a randomly perturbed version of the true gradient, say, as:

$$\widehat{\nabla_w} J_k(w) \triangleq \nabla_w J_k(w) + \boldsymbol{v}_{k,i}(w) \tag{30}$$

where $\boldsymbol{v}_{k,i}(\cdot)$ is the perturbation vector (or gradient noise). Observe that once we replace $\nabla_w J_k(w)$ by $\widehat{\nabla_w} J_k(w)$, then the variables ϕ, ψ, ζ, and w in the diffusion strategies (25a)–(25c) and (26a)–(26c) become random variables (and will be denoted by boldface letters from now on) due to the presence of the random perturbation $\boldsymbol{v}_{k,i}(\cdot)$.

Since the iterate $\boldsymbol{w}_{k,i}$ generated by the diffusion strategies is random, we shall measure performance by examining the average distance between $\boldsymbol{w}_{k,i}$ and $w^\star$, i.e., by evaluating

$$\limsup_{i \to \infty} \ \mathbb{E}\|w^\star - \boldsymbol{w}_{k,i}\| \tag{31}$$

Now, using the solution $w^o(\eta, \tau)$ of (14) we can write:

$$\begin{aligned} &\limsup_{i \to \infty} \ \mathbb{E}\|w^\star - \boldsymbol{w}_{k,i}\| \\ &\leq \underbrace{\|w^\star - w^o(\eta,\tau)\|}_{\text{Approximation Error}} + \limsup_{i \to \infty} \ \mathbb{E}\|w^o(\eta,\tau) - \boldsymbol{w}_{k,i}\| \end{aligned} \tag{32}$$

We will see that under some assumptions, the approximation error $\|w^\star - w^o(\eta,\tau)\|$ can be driven to arbitrarily small

values as $\eta \to \infty$ and $\tau \to 0$. Therefore, it is sufficient to characterize the size of the second term in (32) in order to assess how small (31) is.

We now list the assumptions that are used for studying the performance of the diffusion strategies. These conditions are of the same nature as assumptions often used in the broad distributed optimization literature.

Assumption 1 (Feasible problem): Problem (11) is feasible and, therefore, a minimizer $w^\star \in \mathbb{W}$ exists. $\diamondsuit$

We also require the uniqueness of the solutions $w^\star$ of (11) and $w^o(\eta,\tau)$ of (14). In order to guarantee these facts, we introduce the following assumption.

Assumption 2 (Individual costs): Each $J_k(w)$ has a Hessian matrix that is bounded from above, i.e., there exist $\{\lambda_{k,\max} > 0\}$ such that, for each $k = 1,\ldots,N$:

$$\nabla_w^2 J_k(w) \leq \lambda_{k,\max} I_M \tag{33}$$

Furthermore, since the individual costs $J_k(w)$ are strongly convex, there exist $\lambda_{k,\min} > 0$ such that

$$\nabla_w^2 J_k(w) \geq \lambda_{k,\min} I_M \tag{34}$$

$\diamondsuit$

Fact 1 (Uniqueness of $w^\star$): When Assumptions 1–2 hold, the optimizer $w^\star$ of (11) is unique. $\diamondsuit$

Fact 2 (Uniqueness of $w^o(\eta,\tau)$): When Assumption 2 holds, the optimizer $w^o(\eta,\tau)$ of (14) is unique for any $\eta \geq 0$ and $\tau \geq 0$. $\diamondsuit$

We also require the Hessian matrices of the penalty functions with respect to w to be bounded from above, but not necessarily from below (they are obviously nonnegative definite since the penalty functions are convex).

Assumption 3 (Penalty functions): The Hessian matrix of each penalty function $p_k(w,\tau)$, with respect to w, is upper bounded, i.e.,

$$\nabla_w^2 p_k(w,\tau) \leq \lambda_{k,\max}^p I_M \tag{35}$$

where $\lambda_{k,\max}^p > 0$. Furthermore, since the penalty functions are convex, the Hessian matrix is nonnegative definite. $\diamondsuit$

Observe that we are not requiring $\delta^{\mathrm{IP}}(g_{k,l}(w))$, $\delta^{\mathrm{EP}}(h_{k,u}(w))$, or $g_{k,l}(w)$ to be strongly convex in w.

Assumption 4 (Combination matrix): The combination matrix A in the penalized ATC or CTA implementation is primitive and doubly-stochastic. $\diamondsuit$

A doubly-stochastic matrix A is one that satisfies $A^{\mathsf{T}}\mathbb{1} = \mathbb{1}$ and $A\mathbb{1} = \mathbb{1}$. The primitive condition on A is satisfied by any connected network with at least one self-loop (i.e., at least one $a_{k,k} > 0$) [20].

Assumption 5 (Gradient noise model): Let the symbol $\boldsymbol{\phi}$ refer to either $\boldsymbol{w}$ in the ATC implementation (25a)–(25b) or $\boldsymbol{\psi}$ in the CTA implementation (26a)–(26c). We model the perturbed gradient vector as:

$$\widehat{\nabla_w} J_k(\boldsymbol{\phi}) = \nabla_w J_k(\boldsymbol{\phi}) + \boldsymbol{v}_{k,i}(\boldsymbol{\phi}) \tag{36}$$

where, conditioned on the past history of the estimators $\mathcal{H}_{i-1} \triangleq \{\boldsymbol{\phi}_{k,j} : k = 1,\ldots,N \text{ and } j \leq i-1\}$, the gradient noise $\boldsymbol{v}_{k,i}(\boldsymbol{\phi})$ satisfies:

$$\mathbb{E}\{\boldsymbol{v}_{k,i}(\boldsymbol{\phi})|\mathcal{H}_{i-1}\} = 0 \tag{37}$$

$$\mathbb{E}\|\boldsymbol{v}_{k,i}(\boldsymbol{\phi})\|^2 \leq \alpha\mathbb{E}\|w^o(\eta,\tau) - \boldsymbol{\phi}\|^2 + \sigma_v^2 \tag{38}$$

for some $\alpha \geq 0$, $\sigma_v^2 \geq 0$, and where $\boldsymbol{\phi} \in \mathcal{H}_{i-1}$. $\diamondsuit$

V. MAIN RESULTS

For convenience, we introduce the compact notation:

$$w^o(\infty,\tau) \triangleq \lim_{\eta\to\infty} w^o(\eta,\tau) \tag{39}$$

where $\eta \to \infty$ monotonically.

Theorem 1 (Approaching optimal solution): Under Assumptions 1 and 2, it holds that:

$$\|w^\star - w^o(\infty,0)\| = 0 \tag{40}$$

so that $w^o(\infty,0)$ is feasible and optimal.

Proof: Omitted for brevity — see also [21, p. 477]. $\diamondsuit$

Result (40) is consistent with traditional results from optimization theory [19, p. 288], [22, p. 302–307] where η is replaced by an iteration-dependent sequence, $\eta(i)$, that is required to be divergent.

Theorem 2 (Convergence condition): Let Assumptions 2, 3, 4, and 5 hold. Then, the diffusion algorithm converges for sufficiently small positive step-sizes, namely, for step-sizes that satisfy

$$\mu < \min_{1\leq k\leq N} \left\{ \frac{2\lambda_{k,\max}}{\lambda_{k,\max}^2 + 4\alpha\lambda_{k,\max}^2}, \frac{2\lambda_{k,\min}}{\lambda_{k,\min}^2 + 4\alpha\lambda_{k,\max}^2}, \frac{2}{\eta\cdot(L_k+U_k)\lambda_{k,\max}^p} \right\} \tag{41}$$

Specifically, it holds that for small μ:

$$\limsup_{i\to\infty} \ \mathbb{E}\|w^o(\eta,\tau) - \boldsymbol{w}_{k,i}\| \leq O(\sqrt{\mu}) + O(\eta\cdot\mu) \tag{42}$$

so that

$$\lim_{\mu\to 0} \limsup_{i\to\infty} \ \mathbb{E}\|w^o(\eta,\tau) - \boldsymbol{w}_{k,i}\| = 0. \tag{43}$$

Proof: Omitted for brevity. $\diamondsuit$

We conclude from Theorems 1 and 2 that if we choose the parameters η and τ in terms of μ as follows:

$$\eta \triangleq \mu^{-\theta_1}, \quad \tau = \mu^{\theta_2}, \quad 0 < \theta_1 < 1, \quad 0 < \theta_2 \tag{44}$$

then it holds that

$$\lim_{\mu\to 0} \limsup_{i\to\infty} \ \mathbb{E}\|w^\star - \boldsymbol{w}_{k,i}\| = 0 \tag{45}$$

In addition, the diffusion algorithm, which utilizes a constant step-size, is capable of tracking varying constraint sets.

REFERENCES

[1] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 221–229, Apr. 2013.

[2] K. Srivastava and A. Nedic, "Distributed asynchronous constrained stochastic optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 772–790, Aug. 2011.

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

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

[5] S. Kar, J. Moura, and K. Ramanan, "Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication," *IEEE Transactions on Information Theory*, vol. 58, no. 6, pp. 3575–3605, Jun. 2012.

[6] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections," *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97–123, Jan. 2011.

[7] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[8] 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, Jan. 2011.

[9] A. Nedic and A. Ozdaglar, "Cooperative distributed multi-agent optimization," in *Convex optimization in Signal Processing and Communications*, D. P. Palomar and Y. C. Eldar, Eds. Cambridge University Press, NY, 2010.

[10] F. Yan, S. Sundaram, S. Vishwanathan, and Y. Qi, "Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties," to appear in *IEEE Transactions on Knowledge and Data Engineering.* Also available as *arXiv:1006.4039v3*, Feb. 2011.

[11] S. S. Ram, A. Nedic, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 516–545, Jul. 2010.

[12] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 8, pp. 4289–4305, Aug. 2012.

[13] ——, "Distributed Pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, Apr. 2013.

[14] S. Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6217–6234, Dec. 2012.

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

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

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

[18] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM Journal on Optimization*, vol. 7, no. 4, pp. 913–926, Nov. 1997.

[19] B. T. Polyak, *Introduction to Optimization*. Optimization Software, NY, 1987.

[20] A. H. Sayed, "Diffusion adaptation over networks," in *E-Reference Signal Processing,* R. Chellapa and S. Theodoridis, editors, Elsevier, 2013. Also available as *arXiv:1205.4220v1*, May 2012.

[21] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, NJ, 2006.

[22] D. G. Luenberger, *Optimization by Vector Space Methods*. Wiley, NJ, 1997.