Distributed Asynchronous Deterministic and Stochastic Gradient Optimization Algorithms

JOHN N. TSITSIKLIS, MEMBER, IEEE, DIMITRI P. BERTSEKAS, FELLOW, IEEE, AND MICHAEL ATHANS, FELLOW, IEEE

Abstract—We present a model for asynchronous distributed computation and then proceed to analyze the convergence of natural asynchronous distributed versions of a large class of deterministic and stochastic gradient-like algorithms. We show that such algorithms retain the desirable convergence properties of their centralized counterparts, provided that the time between consecutive interprocessor communications and the communication delays are not too large.

Introduction

MANY deterministic and stochastic iterative algorithms admit a natural distributed implementation [1], [3]–[5], [7] whereby several processors perform computations and exchange messages with the end-goal of minimizing a certain cost function. If all processors communicate to each other their partial results at each instance of time and perform computations synchronously, the distributed algorithm is mathematically equivalent to a single processor (serial) algorithm and its convergence may be studied by conventional means. Synchronous algorithms may have, however, certain drawbacks, which have been discussed in [9].

In this paper we study asynchronous distributed iterative optimization algorithms in which each processor does not need to communicate to each other processor at each time instance; also, processors may keep performing computations without having to wait until they receive the messages that have been transmitted to them; processors are allowed to remain idle some of the time; finally, some processors may perform computations faster than others. Such algorithms can alleviate communication overloads and they are not excessively slowed down by neither communication delays, nor by differences in the time it takes processors to perform one computation.

In Section II we present the model of distributed computation to be employed. In this model, there is a number of processors who perform certain computations and update some of the components of a vector stored in their memory. In the meanwhile, they exchange messages, thus informing each other about the results of their latest computations. Processors who receive messages use them either to update directly some of the components of the vector in their memory, or they may combine the message with the outcome of their own computations, by forming a convex combination. Weak assumptions are made about the relative timing and frequency of computations or message transmissions by the processors.

In Section III we employ this model of computation and also assume that the (possibly random) updates of each processor are gradient-like; that is, they are expected to be in a descent direction, when conditioned on the past history of the algorithm. Our main results show that, under certain assumptions, asynchronous distributed algorithms have similar convergence properties as their centralized counterparts, provided that the time between consecutive communications between processors plus communication delays are not too large. We distinguish two cases: a) constant step-size algorithms (e.g., deterministic gradient-type algorithms) in which the time between consecutive communications has to be bounded for convergence to be guaranteed and; b) decreasing step-size algorithms (e.g., stochastic approximation-type algorithms) for which convergence is proved even if the time between consecutive communications increases without bound as the algorithm proceeds. Sections II and III are developed in parallel with a variety of examples which are used to motivate and explain the formal assumptions that are being introduced.

Finally, Section IV suggests some extensions and possible applications. The Appendix contains the proofs of our main results.

II. A Model of Distributed Computation

We present here the model of distributed computation employed in this paper. We also define the notation and conventions to be followed. Related models of distributed computation have been used in [3]–[5], [7], in which each processor specialized in updating a different component of some vector. The model developed here is more general, in that it allows different processors to update the same component of some vector. If their individual updates are different, their disagreement is (asymptotically) eliminated through a process of communicating and combining their individual updates. In such a case, we will say that there is *overlap* between processors. Another minor difference is that [3] and [7] assumed a shared memory model, whereas we assume that each processor has its own local memory.

Let $H_1, H_2, \cdots, H_L$ be finite-dimensional real vector spaces[1] and let $H = H_1 \times H_2 \times \cdots \times H_L$, which we endow with the Euclidean norm. If $x = (x_1, x_2, \cdots, x_L)$, $x_l \in H_l$, we will refer to x_l as the lth component of x.

Let $\{1, \cdots, M\}$ be the set of processors that participate in the distributed computation. As a general rule concerning notation, we use subscripts to indicate a component of an element of H, superscripts to indicate an associated processor; we indicate time by an argument that follows.

The algorithms to be considered evolve in discrete time. Even if a distributed algorithm is asynchronous and communication delays are real (i.e., not integer) variables, the events of interest (an update by some processor, transmission or reception of a message) may be indexed by a discrete variable; so, the restriction to discrete time entails no loss of generality.

It is important here to draw a distinction between "global" and "local" time. The time variable we have just referred to corresponds to a global clock. Such a global clock is needed only

Manuscript received December 19, 1984; revised October 25, 1985. This paper is based on a prior submission of February 18, 1984. Paper recommended by Past Associate Editor, J. Walrand. This work was supported by the Office of Naval Research under Grants N00014-77-C-0532 and N00014-84-K-0519 (NR 649-003) and by the National Science Foundation under Grant ECS-8217668.

The authors are with the Laboratory for Information and Decision Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139.

IEEE Log Number 8609873.

[1] All of our results generalize to the case where H_l is a Banach space. No modifications whatsoever are needed in the assumptions or the proofs except that matrices should be now called linear operators and that expressions like $\nabla J(x)$ should be interpreted as elements of the dual of H rather than vectors in H.

0018-9286/86/0900-0803$01.00 © 1986 IEEE

for analysis purposes. On the other hand, the processors may be working without having access to a global clock. They may have access to a local clock or to no clock at all.

We assume that each processor has a buffer in its memory in which it keeps some element of H. The value stored by the ith processor at time n (global) is denoted by $x^i(n)$. At time n, each processor may receive some exogenous measurements and/or perform some computations. This allows it to compute a "step" $s^i(n) \in H$, to be used in evaluating the new vector $x^i(n+1)$. Besides their own measurements and computations, processors may also receive messages from other processors, which will be taken into account in evaluating their next vector. The process of communications is assumed to be as follows.

At any time n, processor i may transmit some (possibly all) of the components of $x^i(n)$ to some (possibly all or none) of the other processors. (In a physical implementation, messages do not need to go directly from their origin to their destination; they may go through some intermediate nodes. Of course, this does not change the mathematical model presented here.) We assume that communication delays are bounded. For convenience, we also assume that for any pair (i, j) of processors, for any component x_l and any time n, processor i may receive at most one message originating from processor j and containing an element of H_l. This leads to no significant loss of generality: for example, a processor that receives two messages simultaneously could keep only the one which was most recently sent; if messages do not carry time-stamps, there could be some other arbitration mechanism. Physically, of course, simultaneous receptions are impossible; so, a processor may always identify and keep the most recently received message, even if all messages arrived at the same discrete time n.

If a message from processor j, containing an element of H_l, is received by processor i $(i \neq j)$ at time n, let $t_l^{ij}(n)$ denote the time that this message was sent. Therefore, the content of such a message is precisely $x_l^j(t_l^{ij}(n))$. Naturally, we assume that $t_l^{ij}(n) \leq n$. For notational convenience, we also let $t_l^{ii}(n) = n$, for all i, l, n. We will be assuming that the algorithm starts at time 1; accordingly, we assume that $t_l^{ij}(n) \geq 1$. Finally, we denote by T_l^{ij} the set of all times that processor i receives a message from processor j, containing an element of H_l. To simplify matters we will assume that, for any i, j, l, the set T_l^{ij} is either empty or infinite.

Once processor i has received the messages arriving at time n and has also evaluated $s^i(n)$, it evaluates its new vector $x^i(n+1) \in H$ by forming (componentwise) a convex combination of its old vector and the values in the messages it has just received, as follows:

$$x_l^i(n+1) = \sum_{j=1}^{M} a_l^{ij}(n) x_l^j(t_l^{ij}(n)) + \gamma^i(n) s_l^i(n), \qquad n \geq 1 \quad (2.1)$$

where $s_l^i(n)$ is the lth component of $s^i(n)$ and the coefficients $a_l^{ij}(n)$ are scalars satisfying

$$\text{i)}\ a_l^{ij}(n) \geq 0 \qquad \forall\, i, j, l, n, \quad (2.2)$$

$$\text{ii)}\ \sum_{j=1}^{M} a_l^{ij}(n) = 1, \qquad \forall\, i, l, n, \quad (2.3)$$

$$\text{iii)}\ a_l^{ij}(n) = 0, \qquad \forall\, n \notin T_l^{ij},\ i \neq j. \quad (2.4)$$

Remarks:

1) Note that $t_l^{ij}(n)$ has been defined only for those times n that processor i receives a message of a particular type, i.e., for $n \in T_l^{ij}$. However, whenever $t_l^{ij}(n)$ is undefined, we have assumed above that $a_l^{ij}(n) = 0$, so that (2.1) has an unambiguous meaning.

2) When we refer to a processor performing a "*computation*," we mean the evaluation and addition of the term $\gamma^i(n)s_l^i(n)$ in (2.1). With this terminology, forming the convex combination in (2.1) is not called a computation. We denote by T_l^i the set of all times that processor i performs a computation involving the lth component. Whenever $n \notin T_l^i$, it is understood that $s_l^i(n)$ in (2.1) equals zero. We assume again that for any i, l the set T_l^i is either infinite or empty. Accordingly, processor i will be called *computing*, or *noncomputing, for component l.*

3) The quantities $\gamma^i(n)$ in (2.1) are nonnegative scalar step-sizes. These step sizes may be constant (e.g., $\gamma^i(n) = \gamma_0$, $\forall n$), or time-varying, e.g., $\gamma^i(n) = 1/t_n^i$, where t_n^i is the number of times that processor i has performed a computation up to time n. Notice that with the latter choice each processor may evaluate its step size using only a local counter rather than a global clock.

Examples

We now introduce a collection of simple examples representing various classes of algorithms we are interested in, so as to illustrate the nature of the assumptions to be introduced later. Throughout, we assume that communication delays are bounded. We actually start with a broad classification and then proceed to more special cases. In these examples, we model the message receptions and transmissions [i.e., the sets T_l^{ij} and the variables $t_l^{ij}(n)$], the times at which computations are performed (i.e., the sets T_l^i) and the combining coefficients $a_l^{ij}(n)$ as *deterministic.* (This does not mean, however, that they have to be *a priori* known by the processors.)

Specialization: This is the case considered in [4], [5] where each processor updates a particular component of the x-vector specifically assigned to it and relies on messages from the other processors for the remaining components. Formally

i) $M = L$. (There are as many processors as there are components.)

ii) $s_l^i(n) = 0$, $\forall l \neq i$, $\forall n$. (A processor may update only its own component; $T_l^i = \phi$, $\forall i \neq l$.)

iii) Processor j only sends messages containing elements of H_j; if processor i receives such a message, it uses it to update x_j^i by setting x_j^i equal to the value received. Equivalently,

a) If $i \neq j$ and $j \neq l$, then $T_l^{ij} = \phi$ and $a_l^{ij}(n) = 0$, $\forall n$.

b) If processor i receives a message from processor j at time n, i.e., if $n \in T_j^{ij}$, then $a_j^{ij}(n) = 1$. Otherwise, $a_j^{ij}(n) = 0$, and $a_j^{ii}(n) = 1$.

Overlap: Here $L = 1$ (we do not distinguish components of elements of H), messages contain elements of H (not just components) and each processor may update any component of x. (For this case subscripts are redundant and will be omitted.)

We now assume that $J{:}H \rightarrow [0, \infty)$ is a continuously differentiable nonnegative cost function with a Lipschitz continuous derivative.

Example I—Deterministic Gradient Algorithm; Specialization: Let $\gamma^i(n) = \gamma_0 > 0$, $\forall n, i$. At each time $n \in T_i^i$ that processor i updates x_i^i, it computes $s_i^i(n) = -\partial J/\partial x_i(x^i(n))$ and lets $s_j^i(n) = 0$, for $j \neq i$. We assume that each processor i communicates its components x_i^i to every other processor at least once every B_1 time units, for some constant B_1. Other than this restriction, we allow the transmission and reception times to be arbitrary. (A related stochastic algorithm could be obtained by letting $s_i^i(n) = -\partial J/\partial x_i(x^i(n))(1 + w_i^i(n))$, where $w_i^i(n)$ is unit variance white noise, independent for different i's.)

Example II—Newton's Method; Overlap: For simplicity we assume that there are only two processors ($M = 2$). Let $\gamma^i(n) = \gamma_0 > 0$, $\forall n$. We also assume that J is twice continuously differentiable, strictly convex and its Hessian matrix, denoted by $G(x)$, satisfies $0 < \delta_1 I \leq G(x) \leq \delta_2 I$, $\forall x \in H$. At each time $n \in T^i$, processor i computes $s^i(n) = -G^{-1}(x^i(n))\partial J/\partial x(x^i(n))$. For $n \notin T^i$, $s^i(n) = 0$. If at time n processor 1 (respectively, 2) receives a message $x^2(t^{12}(n))$ [respectively, $x^1(t^{21}(n))$], it updates its vector by $x^1(n+1) = a_{11}x^1(n) + a_{12}x^2(t^{12}(n)) + \gamma^1(n)s^1(n)$, [respectively, $x^2(n+1) = a_{21}x^1(t^{21}(n)) + a_{22}x^2(n) + \gamma^2(n)s^2(n)$]. Here we assume that $0 < a_{ij} < 1$ and that $a_{11} + a_{12}$

$= a_{21} + a_{22} = 1$. For other times n the same formula is used with $a_{12} = 0$ ($a_{21} = 0$). We make the same assumptions on transmission and reception times as in Example 1.

Example III—Distributed Stochastic Approximation; Specialization: Let $\gamma^i(n)$ be such that, for some positive constants A_1, A_2, $A_1/n \leq \gamma^i(n) \leq A_2/n$, $\forall n$. Notice that the implementation of such a step size only requires a local clock that runs in the same time scale (i.e., within a constant factor) as the global clock. For $n \in T^i$, let $s_i^i(n) = -\partial J/\partial x_i(x^i(n)) + w_i^i(n)$. Also, $s_j^i(n) = 0$, for $i \neq j$ and for all n. We assume that $w_i^i(n)$, conditioned on the past history of the algorithm has zero mean and that $E[\|w_i^i(n)\|^2|x^i(n)] \leq K(\|\nabla J(x^i(n))\|^2 + 1)$, for some constant K. We assume that for some $B_1 \geq 0$, $\beta \geq 1$ and for all n, each processor communicates its component x_i^i to every other processor at least once during the time interval $[B_1 n^\beta, B_1(n + 1)^\beta]$. Other than the above restriction, we allow transmission and reception times to be arbitrary. Notice that the above assumptions allow the time between consecutive communications to grow without bound.

Example IV—Distributed Stochastic Approximation: Overlap: Let $\gamma^i(n)$ be as in Example III and let $M = 2$. For $n \in T^i$, let $s^i(n) = -\partial J/\partial x(x^i(n)) + w^i(n)$, where $w^i(n)$ is as in Example III. We make the same assumptions on transmission and reception times as in Example III. Whenever a message is received, a processor combines its vector with the content of that message using the combining rules of Example II.

Example V: This example is rather academic but will serve to illustrate some of the ideas to be introduced later. Consider the case of overlap, assume that H is one-dimensional, and let $\gamma^i(n) = 1$, $\forall n$. Assume that, at each time n, either all processors communicate to each other, or no processor sends any message. Let the communication delays be zero (so, $t^{ij}(n) = n$, whenever $t^{ij}(n)$ is defined) and assume that $a^{ij}(n) = a^{ij}$ (constant) at those times n that messages are exchanged. We define vectors $x(n) = (x^1(n), \cdots, x^M(n))$ and $s(n) = (s^1(n), \cdots, s^M(n))$. Then, the algorithm (2.1) may be written as

$$x(n+1) = A(n)x(n) + s(n). \tag{2.5}$$

For each time n, either $A(n) = I$ (no communications) or $A(n) = A$, the matrix consisting of the coefficients a^{ij}. The latter is a "stochastic" matrix: it has nonnegative entries and each row sums to 1. We assume that each a^{ij} is positive. It follows that $\bar{A} = \lim_{n\to\infty} A^n$ exists and has identical rows with positive elements. We assume that the time between consecutive communications is bounded but otherwise arbitrary. Clearly then, $\lim_{n\to\infty} \Pi_{m=k}^n A(m) = \bar{A}$, for all k. It is interesting to compare (2.5) with the generic equation

$$x(n+1) = x(n) + s(n)$$

which arises in centralized algorithms.

All of our examples refer to either specialization or overlap. However, we may also conceive of situations in which some of the components are updated by a single processor, while some others are updated by many processors simultaneously (partial overlap).

Assumptions on the Communications and the Combining Coefficients

We now consider a set of assumptions on the nature of the communication and combining process, so that the preceding examples appear as special cases.

For each component $l \in \{1, \cdots, L\}$ we introduce a directed graph $G_l = (V, E_l)$ with nodes $V = \{1, \cdots, M\}$ corresponding to the set of processors. An edge (j, i) belongs to E_l if and only if T_l^{ij} is infinite, that is, if and only if processor j sends (in the long run) an infinite number of messages to processor i with a value of the lth component x_l^j.

Assumption 2.1: For each component $l \in \{1, \cdots, L\}$, the following hold.

a) There is at least one computing processor for component l.

b) There is a directed path in G_l, from every computing processor (for component l) to every other processor (computing or not).

c) There is some $\alpha > 0$ such that:

i) If processor i receives a message from processor j at time n (i.e., if $n \in T_l^{ij}$), then $a_l^{ij}(n) \geq \alpha$.

ii) For every computing processor i, $a_l^{ii}(n) \geq \alpha$, $\forall n$.

iii) If processor i is noncomputing and has in-degree[2] (in G_l) larger than or equal to 2, then $a_l^{ii}(n) \geq \alpha$, $\forall n$.

Parts b) and c) of Assumption 2.1 guarantee that any update by any computing processor has a lasting effect on the states of computation of all other processors. Next, we introduce two alternative assumptions on the frequency of communications.

Assumption 2.2: The time between consecutive transmissions of component x_l^j from processor j to processor i is bounded by some $B_1 \geq 0$, for all $(j, i) \in E_l$.

Assumption 2.3: There are constants $B_1 > 0$, $\beta \geq 1$ such that, for any $(j, i) \in E_l$, and for any n, at least one message x_l^j is sent from processor j to processor i during the time interval $[B_1 n^\beta, B_1(n + 1)^\beta]$. Moreover, the total number of messages transmitted and/or received during any such interval is bounded.

Note that Assumption 2.2 is a special case of 2.3, with $\beta = 1$.

Assumption 2.4: Communication delays are bounded by some $B_0 \geq 0$, i.e., for all i, j, l and $n \in T_l^{ij}$ we have $n - t_l^{ij}(n) \leq B_0$.

Assumptions 2.1 and 2.4 hold for all the examples introduced above. Assumption 2.2 holds for Examples I, II, and V; Assumption 2.3 holds for Examples III and IV, except for its last part which has to be explicitly introduced.

Equation (2.1) which defines the structure of the algorithm is a linear system driven by the steps $s_j^i(n)$. In the special case where communication delays are zero, we have $t_l^{ij}(n) = n$, and (2.1) becomes a linear system with state vector $(x^1(n), \cdots, x^M(n))$. Equation (2.5) of Example V best illustrates this situation. In general, however, the presence of communication delays necessitates an augmented state if a state space representation is desired.[3] Exploiting linearity, we conclude that there exist scalars $\Phi_l^{ij}(n|k)$, for $n \geq k$, such that

$$x_l^i(n) = \sum_{j=1}^{M} \Phi_l^{ij}(n|0)x_l^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi_l^{ij}(n|k)s_l^j(k). \tag{2.6}$$

The coefficients $\Phi_l^{ij}(n|k)$ are determined by the sequence of transmission and reception times and the combining coefficients. Consequently, they are unknown, in general. Nevertheless, they have the following qualitative properties.

Lemma 2.1:

i) $$0 \leq \Phi_l^{ij}(n|k), \qquad \forall\, i, j, l, n \geq k,$$

$$\sum_{j=1}^{M} \Phi_l^{ij}(n|k) \leq 1, \qquad \forall\, i, l, n \geq k.$$

ii) Under Assumptions 2.1 and 2.4 and either Assumption 2.2 or 2.3, $\lim_{n\to\infty} \Phi_l^{ij}(n|k)$ exists, for any i, j, k, l. The limit is independent of i and will be denoted by $\Phi_l^j(k)$. Moreover, there is a constant $\eta > 0$ such that, if j is a computing processor for component l, then

$$\Phi_l^j(k) \geq \eta, \qquad \forall\, k. \tag{2.9}$$

[2] The in-degree of a processor (node) i (in G_l) is the number of edges in E_l pointing to node i.

[3] Such an augmented state should incorporate all messages that have been transmitted but not yet received. Since we are assuming bounded communication delays, there can only be a bounded number of such messages and the augmented system may be chosen finite dimensional.

The constant η, depends only on the constants introduced in our assumptions (i.e., B_0, B_1, β, α).

iii) Under Assumptions 2.1, 2.2, and 2.4, there exist $d \in [0, 1)$, $B \geq 0$ (depending only on B_0, B_1, α) such that

$$\max_{i,i} |\Phi_l^{ij}(n|k) - \Phi_l^j(k)| \leq Bd^{n-k}, \qquad \forall\ l,\ n \geq k. \tag{2.10}$$

iv) Under Assumptions 2.1, 2.3, and 2.4, there exist $d \in [0, 1)$, $\delta \in (0, 1]$, $B \geq 0$ (depending only on B_0, B_1, β, α) such that

$$\max_{i,j}\ |\Phi_l^{ij}(n|k) - \Phi_l^j(k)| \leq Bd^{n^\delta - k^\delta}, \qquad \forall\ l,\ n \geq k. \tag{2.11}$$

The proof of Lemma 2.1 is omitted and may be found in [18]. Apart from part i) [which is proved by a straightforward induction based on (2.1)] the main idea of the proof, for the case of zero transmission delays, is the following: proving convergence of $\Phi(n|k)$ is equivalent to proving convergence of a sequence of products of stochastic matrices. These stochastic matrices have a "scrambling" property and the desired conclusions follow from well-known results on weak ergodicity of nonstationary Markov chains [15]. The general case may be reduced to the zero delay case by a suitable "state augmentation" procedure.

In the light of (2.6), Lemma 2.1 admits the following interpretation: part ii) states that if all processors cease updating (that is if they set $s^i(n) = 0$) from some time on, they will asymptotically converge to a common limit. Moreover, this common limit depends by a nonnegligible factor on all past updates of all computing processors. Parts iii) and iv) quantify the natural relationship between the frequency of interprocessor communications and the speed at which agreement is reached.

For any pair (i, j) of processors, we define a linear transformation $\Phi^{ij}(n|k): H \to H$ by

$$\Phi^{ij}(n|k)x = (\Phi_1^{ij}(n|k)x_1, \cdots, \Phi_L^{ij}(n|k)x_L) \tag{2.12}$$

where $x = (x_1, \cdots, x_L)$. Clearly, $\lim_{n\to\infty} \Phi^{ij}(n|k)$ exists, is independent of i and will be denoted by $\Phi^j(k)$.

We can now define a vector $y(n) \in H$ by

$$y(n) = \sum_{j=1}^{M} \Phi^j(0)x^j(1) + \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma^j(k)\Phi^j(k)s^j(k) \tag{2.13}$$

and note that $y(n)$ is recursively generated by

$$y(n+1) = y(n) + \sum_{j=1}^{M} \gamma^j(n)\Phi^j(n)s^j(n). \tag{2.14}$$

The vector $y(n)$ is the element of H at which all processors would asymptotically agree if they were to stop computing (but keep communicating and combining) at a time n. It may be viewed as a concise global summary of the state of computation at time n, in contrast to the vectors $x^i(n)$ which are the local states of computation. Another reason for introducing $y(n)$ is that (2.14) is much simpler than (2.1). The content of the vector $y(n)$ and of the $\Phi^j(n)$'s is easiest to visualize in two special cases.

Specialization (e.g., Examples I and III): Here $y(n)$ takes each component from the processor who specializes in that component. That is, $y(n) = (x_1^1(n), \cdots, x_M^M(n))$. Accordingly, $\Phi_j^i(n) = 0$, for $i \neq j$, and $\Phi_j^j(n) = 1$.

Example V: Here $\Phi^{ij}(n|k)$ is the ijth entry of the matrix $\Pi_{m=k+1}^{n-1} A(m)$. It follows that the limit of $\Phi^{ij}(n|k)$ is the ijth entry of $\bar{A}$, which by our assumptions depends only on j. Moreover, $y(n)$ equals any component of $\bar{A}x(n)$. (All components are equal by our assumptions.) If we multiply both sides of (2.5) by $\bar{A}$ and note that $\bar{A}A(n) = \bar{A}$, we obtain $y(n+1) = y(n) + \Sigma_{j=1}^M \bar{A}_{ij}s^j(n)$, which is precisely (2.14).

III. Convergence Results

There is a large number of well-known centralized deterministic and stochastic optimization algorithms which have been analyzed using a variety of analytical tools [2], [10], [11], [14]. A large class of them, the so-called "pseudogradient" algorithms [14], have the distinguishing feature that the (expected) direction of update (conditioned upon the past history of the algorithm) is a descent direction with respect to the cost function to be minimized. The Examples of Section II certainly have such a property. Reference [14] presents a larger list of examples. In this section we present convergence results for the natural distributed asynchronous versions of pseudogradient algorithms. We adopt the model of computation and the corresponding notation of Section II.

We allow the initialization $\{x^1(1), \cdots, x^M(1)\}$ of the algorithm to be random, with finite mean and variance. We also allow the updates $s^i(n)$ of each processor to be random. On the other hand, we assume that $\gamma^i(n)$ is deterministic; we also model the combining coefficients $a_l^{ij}(n)$ and the sequence of transmission and reception times as being deterministic. This is not a serious restriction because they do not need to be known by the processors in advance in order to carry out the algorithm. We assume that all random variables of interest are defined on a probability space (Ω, F, P). We introduce $\{F_n\}$, an increasing sequence of σ-fields contained in F and describing the history of the algorithm up to time n. In particular, F_n is defined as the smallest σ-field such that $s^i(k)$, $k \leq n-1$, and $x^i(1)$, $i \in \{1, \cdots, M\}$ are F_n-measurable.

We assume that the objective of the algorithm is to minimize a nonnegative cost function $J: H \to [0, \infty)$.

Assumption 3.1: J is continuously differentiable and its derivative satisfies the Lipschitz condition

$$\|\nabla J(x) - \nabla J(x')\| \leq K\|x - x'\|, \qquad \forall\ x,\ x' \in H \tag{3.1}$$

where K is some nonnegative constant.

Assumption 3.2: The updates $s_l^i(n)$ of each processor satisfy

$$E\left[\frac{\partial J}{\partial x_l}(x^i(n))s_l^i(n)|F_n\right] \leq 0, \qquad \text{a.s.}\ \ \forall\ i,\ l,\ n.^4$$

This assumption states that each component of each processor's updates is in a descent direction, when conditioned on the past history of the algorithm and it is satisfied by Examples I–IV.

The next assumption is easily seen to hold for Examples I and II. For stochastic algorithms, it requires that the variance of the updates (and, hence, of any noise contained in them) goes to zero, as the gradient of the cost function goes to zero.

Assumption 3.3: For some $K_0 \geq 0$ and for all i, l, n,

$$E[\|s_l^i(n)\|^2] \leq -K_0 E\left[\frac{\partial J}{\partial x_l}(x^i(n))s_l^i(n)\right]$$

As a matter of verifying Assumption 3.3, one would typically check the validity of the slightly stronger condition

$$E[\|s_l^i(n)\|^2|x^i(n)] \leq \quad K_0 E\left[\frac{\partial J}{\partial x_l}(x^i(n))s_l^i(n)|x^i(n)\right]$$

Also, using Lemma 2.1 ii) and Assumption 3.3 we obtain

$$\begin{aligned} E[\|s^i(n)\|^2] &\leq \sum_{l=1}^{L} E[\|s_l^i(n)\|^2] \\ &\leq -\frac{K_0}{\eta} \sum_{l=1}^{L} E\left[\frac{\partial J}{\partial x_l}(x^i(n))\Phi_l^i(n)s_l^i(n)\right] \\ &= -\bar{K}_0 E[\nabla J(x^i(n))\Phi^i(n)s^i(n)] \end{aligned} \tag{3.3}$$

where $\bar{K}_0 = K_0/\eta \geq 0$.

Our first convergence result states that the algorithm converges in a suitable sense, provided that the step sizes employed by each processor are small enough and that the time between consecutive

[4] In (3.2), if H_l has dimension larger than 1, $\partial J/\partial x_l$ should be interpreted as a row vector. In general, the appropriate interpretation should be clear from the context.

communications is bounded, and applies to Examples I and II. It should be noted, however, that Theorem 3.1 (as well as Theorem 3.2 later) does not yet prove convergence to a minimum or a stationary point of J. In particular, there is nothing in our assumptions that prohibits having $s^i(n) = 0$, $\forall i, n$. Optimality is obtained later, using a few auxiliary and fairly natural assumptions (see Corollary 3.1).

Theorem 3.1: Let Assumptions 2.1, 2.2, 2.4, 3.1, 3.2, and 3.3 hold. Suppose also that $\gamma^i(n) \geq 0$ and that $\sup_{i,n} \gamma^i(n) = \gamma_0 < \infty$. There exists a constant $\gamma^* > 0$ (depending on the constants introduced in the Assumptions) such that the inequality $0 < \gamma_0 \leq \gamma^*$ implies the following.

a) $J(x^i(n))$, $i = 1, 2, \cdots, M$, as well as $J(y(n))$, converge almost surely, and to the same limit.

b) $\lim_{n\to\infty} (x^i(n) - x^j(n)) = \lim_{n\to\infty} (x^i(n) - y(n)) = 0$, $\forall i, j$, almost surely and in the mean square.

c) The expression

$$\sum^{\infty} \sum^{M} \gamma^i(n) \nabla J(x^i(n)) E[s^i(n) | F_n] \tag{3.4}$$

is finite, almost surely. Its expectation is also finite.

Proof: See the Appendix.

Theorem 3.1 (as well as Theorem 3.2) is a distributed version of the convergence results of [14] and our proofs follow the same general pattern as in [14]. However, much more technical development is needed to obtain bounds on the effects of asynchronism and therefore show that asynchronism cannot destroy convergence.

The main reason why such results are possible is the following. The difference between $y(n)$ and $x^i(n)$, for any i, is of the order of $A\gamma_0$, where A is proportional to a bound on communication delays plus the time between consecutive communications between processors. Therefore, as long as γ_0 remains small, $\nabla J(x^i(n))$ is approximately equal to $\nabla J(y(n))$; hence $s_l^i(n)$ [and consequently $\Phi^i(n) s^i(n)$] is approximately in a descent direction, starting from point $y(n)$. Therefore, iteration (2.14) is approximately the same as a centralized descent (pseudogradient) algorithm which is, in general, convergent [14]. This line of reasoning is actually reflected in our proofs.

Decreasing Step-Size Algorithms

We now introduce an alternative set of assumptions. We allow the magnitude of the updates $s^i(n)$ to remain nonzero, even if $\nabla J(x^i(n))$ is zero (Examples III and IV). Such situations are common in stochastic approximation algorithms or in system identification applications. Since the noise is persistent, the algorithm can be made convergent only by letting the step size $\gamma^i(n)$ decrease to zero. The choice $\gamma^i(n) = 1/n$ is most commonly used in centralized algorithms and in the sequel we will assume that $\gamma^i(n)$ behaves like $1/n$.

Since the step size is decreasing, the algorithm becomes progressively slower as $n \to \infty$. This allows us to let the communications process become progressively slower as well, provided that it remains fast enough, when compared to the natural time scale of the algorithm, the latter being determined by the rate of decrease of the step size. Such a possibility is captured by Assumption 2.3.

The next assumption, intended to replace Assumption 3.3, allows the noise to be persistent. It holds for Examples I–IV. As in Assumption 3.3, inequality (3.5) could be more naturally stated in terms of conditional expectations, but such a stronger version turns out to be unnecessary.

Assumption 3.4: For some $K_1, K_2 \geq 0$, and for all i, l, n,

$$E[\|s_l^i(n)\|^2] \leq -K_1 E\left[\frac{\partial J}{\partial x_l}(x^i(n)) s_l^i(n)\right] + K_2. \tag{3.5}$$

Theorem 3.2: Let Assumptions 2.1, 2.3, 2.4, 3.1, 3.2, and 3.4, hold and assume that for some $K_3 \geq 0$, $\gamma^i(n) \leq K_3/n$, $\forall n, i$. Then, conclusions a), b), c), of Theorem 3.1 remain valid.

Proof: See the Appendix.

Theorem 3.2 remains valid if (3.5) is replaced by the weaker assumption

$$E[\|s^i(n)\|^2] \leq K_0 E[J(x^i(n))] - K_1 E[\nabla J(x^i(n)) \Phi^i(n) s^i(n)] + K_2. \tag{3.6}$$

The proof may be found in [18] and is significantly more complicated.

We continue with a corollary which shows that, under reasonable conditions, convergence to a stationary point or a global optimum may be guaranteed. We only need to assume that away from stationary points some processor will make a positive improvement in the cost function. Naturally, we only require the processors to make positive improvements at times when they are not idle.

Corollary 3.1: Suppose that for some $K_4 > 0$, $\gamma^i(n) \geq K_4/n$, $\forall n, i$. Assume that J has compact level sets and that there exist continuous functions $g_l^i: H \to [0, \infty)$ such that

$$\frac{\partial J}{\partial x_l}(x^i(n)) E[s_l^i(n) | F_n] \leq -g_l^i(x^i(n)), \qquad \forall\, n \in T_l^i. \tag{3.7}$$

We define $g: H \to [0, \infty)$ by $g(x) = \Sigma_{i=1}^{M} \Sigma_{l=1}^{L} g_l^i(x)$ and we assume that any point $x \in H$ satisfying $g(x) = 0$ is a stationary point of J. Finally, suppose that the difference between consecutive elements of T_l^i is bounded, for any i, l such that $T_l^i \neq \phi$. Then,

a) Under the Assumptions of either Theorem 3.1 or 3.2,

$$\liminf_{n\to\infty} \|\nabla J(x^i(n))\| = 0, \qquad \forall\, i, \text{ a.s.} \tag{3.8}$$

b) Under the Assumptions of Theorem 3.1 and if (for some $\epsilon > 0$) $\gamma^i(n) \geq \epsilon$, $\forall i, n$, we have

$$\lim_{n\to\infty} \|\nabla J(x^i(n))\| = 0, \qquad \forall\, i, \text{ a.s.} \tag{3.9}$$

and any limit point of $\{x^i(n)\}$ is a stationary point of J.

c) Under the Assumptions of either Theorem 3.1 or 3.2 and if every point satisfying $g(x) = 0$ is a minimizing point of J (this is implicitly assuming that all stationary points of J are minima), then

$$\lim_{n\to\infty} J(x^i(n)) = \inf_{x\in H} J(x).$$

Proof: See the Appendix.

We now discuss the above corollary and apply it to our examples. Our assumption on T_l^i states that, for each component l, the time between successive computations of s_l^i is bounded, for any computing processor i for that component. Such a condition will be always met in practice. The assumption $\gamma^i(n) \geq K_4/n$ may be enforced without the processor having access to a global clock. For example, apart from the trivial case of constant step size, we may let $\gamma^i(n) = 1/t_n^i$, where t_n^i is the number of times, before time n, that processor i has performed a computation.

For Examples I and III, (3.7) holds with $g^i(x)$ a constant multiple of $(\partial J/\partial x_i)^2$; for Examples II and IV, it holds with $g^i(x)$ a constant multiple of $\|\nabla J(x)\|^2$. We may conclude that Corollary 3.1 applies and proves convergence for Examples I–IV.

We close this section by pointing out that our results remain valid if we model the combining coefficients, the transmission and reception times as random variables defined on the same probability space (Ω, F, P), subject to certain restrictions [18]. Notice that such a generalization allows the processors to decide when and where to transmit based on information related to the progress of the algorithm. Finally, Assumptions 2.1–2.4 may be dispensed with as long as the conclusion of Lemma 2.1 may be somehow independently verified.

IV. Extensions and Applications

A main direction along which our results may be extended is in analyzing the convergence of distributed algorithms with decreasing step size and with correlated noise, for which the pseudogradient assumption fails to hold. Such algorithms arise frequently, for example, in system identification. Very few global convergence results are available, even for the centralized case [16]. However, as in the centralized case an ordinary differential equation (ODE) may be associated with such algorithms, which may be used to prove local convergence subject to an assumption that the algorithm returns infinitely often to a bounded region [10], [11], [18].

Another issue, arising in the case of constant step-size algorithms, concerns the choice of a step size which will guarantee convergence. We may trace the steps in the proof of Theorem 3.1 and find some bounds on γ_0 so as to ensure convergence, but these bounds will not be particularly tight. For a version of a distributed deterministic gradient algorithm, tighter bounds have been obtained in [18] which quantify the notion that the frequency of communications between different processors should in some sense reflect the degree of coupling inherent in the optimization problem.

Concerning possible applications, there are three broad areas that come to mind. There is first the area of parallel computation, where an asynchronous algorithm could avoid several types of bottlenecks [9]. Then, there is the area of data communication networks in which there has been much interest for distributed algorithms for routing and flow control [6], [8], [19]. An analysis of a gradient projection method for optimal routing has been carried out in [17]. Finally, certain common algorithms for system identification or adaptive filtering fall into the framework of decreasing step-size stochastic algorithms and our approach may be used for analyzing the convergence of their distributed versions [18]. Our results may not be applicable without any modifications or refinements to such diverse applications areas. Nevertheless, our analysis indicates what kind of results should be expected to hold.

Appendix

This Appendix contains the proofs of the results of Section III.

Remark on Notation: In the course of the proofs in this section, we will use the symbol A to denote *nonnegative* constants which are independent of n, γ_0, $\gamma^i(n)$, $x^i(n)$, $s^i(n)$, etc., but which may depend on the constants introduced in the various assumptions (that is, M, L, K, K_0, B_0, B_1, α, etc.). When A appears in different expressions, or even in different sides of the same equality (or inequality), it will not necessarily represent the same constant. (With this convention, an inequality of the form $A + 1 \leq A$ is meaningful and has to be interpreted as saying that $A + 1$, where A is some constant, is smaller than some other constant, denoted again by A.) This convention is followed so as to avoid the introduction of unnecessarily many symbols.

Proof of Theorem 3.1: Without loss of generality, we will assume that the algorithm is initialized so that $x^i(1) = 0$, $\forall i$. In the general case where $x^i(1) \neq 0$, we may think of the algorithm as having started at time 0, with $x^i(0) = 0$; then, a random update $s^i(0)$ sets $x^i(1)$ to a nonzero value. So, the case in which the processors initially disagree may be easily reduced to the case where they initially agree.

Note that we may define $\bar{s}^i(n) = (\gamma^i(n)/\gamma_0)\, s^i(n)$ and view $\bar{s}^i(n)$ as the new step with step-size γ_0. It is easy to see that Assumptions (3.2) and (3.3) also hold for $\bar{s}^i(n)$. For these reasons, no generality is lost if we assume that $\gamma^i(n) = \gamma_0$, $\forall n$ and this is what

$$b(n) = \sum_{i=1}^{M} \|s^i(n)\| \tag{A.1}$$

and note that

$$b^2(n) \leq M \sum_{i=1}^{M} \|s^i(n)\|^2 \leq Mb^2(n).$$

Using (2.6), (2.13) and Lemma 2.1 iii), we obtain

$$\|y(n) - x^i(n)\| \leq \sum_{k=1}^{n-1} \sum_{j=1}^{M} \gamma_0 \|\Phi^j(k) - \Phi^{ij}(n|k)\| \, \|s^j(k)\|$$
$$\leq A\gamma_0 \sum_{k=1}^{n-1} d^{n-k} b(k). \tag{A.2}$$

From a Taylor series expansion for J we obtain

$$J(y(n+1)) = J\left(y(n) + \gamma_0 \sum_{i=1}^{M} \Phi^i(n)s^i(n)\right)$$
$$\leq J(y(n)) + \gamma_0 \nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n)$$
$$+ A \left\| \gamma_0 \sum_{i=1}^{M} \Phi^i(n)s^i(n) \right\|^2$$
$$\leq J(y(n)) + \gamma_0 \nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n) + A\gamma_0^2 b^2(n). \tag{A.3}$$

Assumption 3.2 is in terms of $\nabla J(x^i(n))$, whereas above we have $\nabla J(y(n))$. To overcome this difficulty, we use the Lipschitz continuity of the derivative of J and invoke (A.2) to obtain

$$\left\| \nabla J(y(n)) \sum_{i=1}^{M} \Phi^i(n)s^i(n) - \sum_{i=1}^{M} \nabla J(x^i(n))\Phi^i(n)s^i(n) \right\|$$
$$\leq A \sum_{i=1}^{M} \|y(n) - x^i(n)\| \, \|s^i(n)\|$$
$$\leq \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} b(k) \sum_{i=1}^{M} \|s^i(n)\| = \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} b(k) b(n)$$
$$\leq \gamma_0 A \sum_{k=1}^{n-1} d^{n-k} [b^2(k) + b^2(n)]. \tag{A.4}$$

Let us define

$$G^i(n) = -\nabla J(x^i(n))\Phi^i(n)s^i(n), \tag{A.5}$$

$$G(n) = \sum_{i=1}^{M} G^i(n), \tag{A.6}$$

and note that Assumption 3.2 implies that $E[G(n)] \geq 0$. We now rewrite inequality (A.3) using (A.4) to replace the derivative term, to obtain

$$J(y(n+1)) \leq J(y(n)) - \gamma_0 G(n) + A\gamma_0^2 b^2(n)$$
$$+ A\gamma_0^2 \sum_{k=1}^{n-1} d^{n-k} [b^2(k) + b^2(n)]$$
$$\leq J(y(n)) - \gamma_0 G(n) + A\gamma_0^2 \sum_{k=1}^{n} d^{n-k} b^2(k).$$

Assumption 3.3 implies that [cf. inequality (3.3)]

$$E[b^2(k)] \le AE[G(k)]. \tag{A.8}$$

Taking expectations in (A.7) and using (A.8) we obtain

$$E[J(y(n+1))] \le E[J(y(n))] - \gamma_0 E[G(n)] + A\gamma_0^2 \sum_{k=1}^{n} d^{n-k} E[G(k)]. \tag{A.9}$$

We then sum (A.9) for different values of n, to obtain

$$0 \le E[J(y(n+1))] \le E[J(y(1))] + \sum_{k=1}^{n} \left[A \frac{\gamma_0^2}{1-d} - \gamma_0\right]$$

We now let $\gamma^* = (1-d)/2A$, where A is the constant of inequality (A.10), and assume that $0 < \gamma_0 \le \gamma^*$. Then, inequality (A.10) implies

$$\gamma_0 \sum_{k=1}^{n} E[G(k)] \le 2E[J(y(1))], \ \forall\, n \tag{A.11}$$

and letting n tend to infinity,

$$\gamma_0 \sum_{k=1}^{\infty} E[G(k)] < \infty. \tag{A.12}$$

By Assumption 3.2, $E[G(k) | F_k] \ge 0$, $\forall k$; we may apply the monotone convergence theorem to (A.12) and obtain

$$E\left[\sum_{k=1}^{\infty} E[G(k) | F_k]\right] = \sum_{k=1}^{\infty} E[G(k)] < \infty \tag{A.13}$$

which implies

$$\sum^{\infty} E[G(k) | F_k] < \infty, \qquad \text{a.s.} \tag{A.14}$$

From (A.14) we obtain

$$\sum_{n=1}^{\infty} \nabla_l J(x^i(n)) E[\Phi_l^i(n) s_l^i(n) | F_n] > -\infty, \qquad \text{a.s.}$$

Now use the fact [Lemma 2.1 ii), inequality (2.9)] that $\Phi_l^i(n) \ge \eta > 0$, for any computing processor i for component l. This implies that

$$\sum_{k=1}^{\infty} \nabla_l J(x^i(n)) E[s_l^i(n) | F_n] > -\infty, \qquad \text{a.s.}$$

and establishes part c) of the theorem.

Lemma A.1: Let $X(n)$, $Z(n)$ be nonnegative stochastic processes (with finite expectation) adapted to $\{F_n\}$ and such that

$$E[X(n+1) | F_n] \le X(n) + Z(n), \tag{A.15}$$

$$\sum_{n=1}^{\infty} E[Z(n)] < \infty. \tag{A.16}$$

Then $X(n)$ converges almost surely, as $n \to \infty$.

Proof of Lemma A.1: By the monotone convergence theorem and (A.16) it follows that $\sum_{n=1}^{\infty} Z_n < \infty$, almost surely. Then, Lemma A.1 becomes the same as Lemma 4.C.1 in [12, p. 453], which in turn is a consequence of the supermartingale convergence theorem [13]. □

Now let A be the constant in the right-hand side of (A.7) and let

$$Z(n) = A\gamma_0^2 E\left[\sum_{k=1}^{n} d^{n-k} b^2(k) | F_n\right] \tag{A.17}$$

Then, $Z(n) \ge 0$ and by (A.8)

$$E[Z(n)] \le A \sum_{k=1}^{n} d^{n-k} E[G(k)]. \tag{A.18}$$

Therefore,

$$\sum_{n=1}^{\infty} E[Z(n)] \le A \sum_{n=1}^{\infty} \sum_{k=1}^{n} d^{n-k} E[G(k)]$$

$$= A \frac{\;}{1-d} \sum_{k=1}^{\infty} E[G(k)] < \infty, \tag{A.19}$$

where the last inequality follows from (A.12). Therefore, $Z(n)$ satisfies (A.16). We take the conditional expectation of (A.7), given F_n. Note that $J(y(n))$ is F_n-measurable and that $E[G(n) | F_n] \ge 0$. Therefore Lemma A.1 applies and $J(y(n))$ converges almost surely.

Using Assumption 3.3 once more, together with (A.12),

$$E\left[\sum_{k=1}^{\infty} b^2(k)\right] \le A \sum_{k=1}^{\infty} E[G(k)] < \infty \tag{A.20}$$

which implies that $b(k)$ converges to zero, almost surely. Recall (A.2) to conclude that $y(n) - x^i(n)$ converges to zero, almost surely. Also, by squaring (A.2), taking expectations and using the fact that $E[b^2(k)]$ converges to zero we conclude that $E[\|y(n) - x^i(n)\|^2]$ also converges to zero, and this proves part b) of the Theorem.

Now, let us use Assumption 3.1 and a second-order expansion of J to obtain, for any $a \in R$

$$0 \le J(x - a\nabla J(x)) \le J(x) - aA_1\|\nabla J(x)\|^2 + a^2 A_2 \|\nabla J(x)\|^2 \tag{A.21}$$

where A_1, A_2 are positive constants not depending on a. Assuming that a was chosen small enough, we may use (A.21) and the nonnegativity of J to conclude

$$\|\nabla J(x)\|^2 \le AJ(x), \qquad \forall\, x \in H. \tag{A.22}$$

Since $J(y(n))$ converges, it is bounded; hence $\nabla J(y(n))$ is also bounded, by (A.22). We then use the fact that $y(n) - x^i(n)$ converges to zero, to conclude that $J(x^i(n)) - J(y(n))$ also converges to zero. This proves part a) and concludes the proof of the theorem. ■

In the following lemma we bound certain infinite series by corresponding infinite integrals. This is justified as long as the integrand cannot change by more than a constant factor between any two consecutive integer points. For notational convenience, we use $c(n|k)$ to denote $d^{n^\delta - k^\delta}$, where d and δ are as in Lemma 2.1 iv).

Lemma A.2: The following hold:

$$\sum_{n=1}^{\infty} \sum_{m=n}^{\infty} \frac{1}{m}\frac{1}{n} c(m|n) < \infty, \tag{A.23}$$

$$\lim_{m\to\infty} \sum_{k=1}^{m} \frac{m}{k^2} c(m|k) = \lim_{m\to\infty} \sum_{k=1}^{m} \frac{1}{k} c(m|k) = 0, \tag{A.24}$$

$$\lim_{m\to\infty} \sum_{k=m}^{\infty} \frac{1}{k} c(k|m)=0. \tag{A.25}$$

Proof of Lemma A.2: Let $t^\delta = y$; then, $t = y^{1/\delta}$ and $dt = (1/\delta)y^{1/\delta-1}\, dy$. Therefore,

$$\int_{t=s}^{\infty} \frac{1}{t} d^{t^\delta-s^\delta}\, dt=\frac{1}{\delta} \int_{y=s^\delta}^{\infty} d^{y-s^\delta} \frac{1}{y}\, dy\le\frac{1}{\delta s^\delta} \int_{y=s^\delta}^{\infty} d^{y-s^\delta}\, dy\le\frac{A}{s^\delta} \tag{A.26}$$

where A does not depend on s. Equation (A.25) follows. Since $(1/s)\, A/s^\delta$ is an integrable function of s, (A.23) follows as well.

The left-hand side of (A.24) is bounded by

$$Am \int_{t=1}^{m} \frac{1}{t^2} d^{m^\delta-t^\delta}\, dt=Am \int_{y=1}^{m^\delta} \frac{1}{y^{2/\delta}} d^{m^\delta-y^\delta} \frac{1}{\delta} y^{(1/\delta)-1}\, dy$$

$$=Am \int_{y=1}^{m^\delta} y^{-(1/\delta)-1}\, d^{m^\delta-y}\, dy\le Am^{-\delta}$$

which converges to zero. The middle term in (A.24) is certainly smaller and converges to zero as well. □

Proof of Theorem 3.2: Using the same arguments as in the proof of Theorem 3.1, we may assume, without loss of generality, that $x^i(1) = 0$ and that $\gamma^i(n) = 1/n$, $\forall i, n$. (Otherwise, we could define $\bar{s}^i(n) = n\gamma^i(n)s^i(n)$.)

We still use $c(n|k)$ to denote $d^{n^\delta-k^\delta}$. We define again $b(n)$, $G^i(n)$, $G(n)$ by (A.1), (A.5), (A.6), respectivaly, as in the proof of Theorem 3.1. Also, let

$$\phi(n|k)=\begin{cases} \dfrac{1}{n^2}+\displaystyle\sum_{m=1}^{n-1} \frac{1}{n}\frac{1}{m} c(n|m), & n=k,\\ \dfrac{1}{n}\dfrac{1}{k} c(n|k), & n>k,\\ 0, & n<k. \end{cases} \tag{A.27}$$

By replicating the steps leading to inequality (A.7) in the proof of Theorem 3.1 and using Lemma 2.1 iv) and (A.27) we obtain, for some $A \ge 0$,

$$J(y(n+1))\le J(y(n))-\frac{1}{n} G(n)+A \sum_{k=1}^{n} \phi(n|k)b^2(k), \qquad \forall\, n. \tag{A.28}$$

Taking expectations in (A.28), we have

$$E[J(y(n+1))]\le E[J(y(n))]-\frac{1}{n} E[G(n)] + A \sum_{k=1}^{n} \phi(n|k)E[b^2(k)], \qquad \forall\, n \tag{A.29}$$

and using Assumption 3.4,

$$E[J(y(n+1))]\le E[J(y(n))]-\frac{1}{n} E[G(n)] + A \sum_{k=1}^{n} \phi(n|k)(E[G(k)]+1). \tag{A.30}$$

We then sum (A.30), for different values of n, to obtain

$$0\le E[J(y(n+1))]\le E[J(y(1))]+A \sum_{m=1}^{n} \sum_{k=1}^{m} \phi(m|k)$$

$$\sum_{m=1}^{n} \left[\sum_{k=m}^{n} \phi(k|m)-\frac{1}{m}\right] E[G(m)]. \tag{A.31}$$

The definitions (A.27) and (A.23) imply that the middle term on the right-hand side of (A.31) is bounded. Moreover, using (A.27),

$$m \sum_{k=m}^{\infty} \phi(k|m)=\frac{1}{m}+\sum_{k=1}^{m-1} \frac{1}{k} c(m|k)+ \sum_{k=m}^{\infty} \frac{1}{k} c(k|m),$$

which converges to zero, as $m \to \infty$, by (A.24) and (A.25). Therefore, for large enough m, $\Sigma_{k=m}^{n}\, \phi(k|m) - 1/m \le -1/2m$. It follows that $E[J(y(n))]$ is bounded. Inequality (A.31) and the above also imply that $\Sigma_{m=1}^{\infty}\, (1/m)E[G(m)] < \infty$ and part c) of the Theorem follows, as in the proof of Theorem 3.1.

We now define

$$Z(n)=E\left[\sum_{k=1}^{n} \phi(n|k)b^2(k) | F_n \right]$$

and note that

$$\sum_{n=1}^{\infty} E[Z(n)]\le A \sum_{k=1}^{\infty} \sum_{n=k}^{\infty} \phi(n|k)[E[G(k)]+1]$$

$$\le A+A \sum_{k=1}^{\infty} \frac{1}{k} E[G(k)]<\infty.$$

Taking conditional expectations in (A.28) (with respect to F_n) and using Lemma A.1, we conclude that $J(y(n))$ converges, almost surely.

We now turn to the proof of part b). Using (3.5) and (A.22), we have

$$E[\|s^i(n)\|^2]\le E\left[2A\nabla J(x^i(n)) \frac{1}{2} s^i(n))\right] +A$$

$$\le 4A^2E[\|\nabla J(x^i(n))\|^2]+\frac{1}{4} E[\|s^i(n)\|^2]+A$$

$$\le AE[J(x^i(n))]+\frac{1}{4} E[\|s^i(n)\|^2]+A, \tag{A.32}$$

which finally implies that

$$E[\|s^i(n)\|^2]\le AE[J(x^i(n))]+A. \tag{A.33}$$

Now, using (A.22) once more,

$$J(x^i(n))-J(y(n))\le\|\nabla J(y(n)\| \cdot \|x^i(n)-y(n)\| +A\, \|x^i(n)-y(n)\|^2$$

$$\le\frac{1}{2}\, \|\nabla J(y(n))\|^2+A\, \|x^i(n)-y(n)\|^2$$

$$\le AJ(y(n))+A\, \|x^i(n)-y(n)\|^2. \tag{A.34}$$

Inequalities (A.33), (A.34) and the boundedness of $E[J(y(n))]$ yield

$$E[b^2(n)]\le A+AE[\|x^i(n)-y(n)\|^2].$$

Similarly with (A.2), we have

$$\|y(n)-x^i(n)\|\le A \sum_{k=1}^{n-1} \frac{1}{k} c(n|k)b(k)$$

and

$$\|y(n)-x^i(n)\|^2\le An \sum_{k=1}^{n-1} \frac{c^2(n|k)}{k^2}\, b^2(k)$$

$$\le An \sum_{k=1}^{n-1} \frac{c(n|k)}{k^2}\, b^2(k).$$

Therefore,

$$E[\|y(n)-x^i(n)\|^2]\le\beta(n)\max_{1\le k<n}E[b^2(k)], \qquad (A.38)$$

where $\beta(n) = An\sum_{k=1}^{n-1}c(n|k)/k^2$ converges to zero, by (A.24). Using (A.35),

$$E[\|y(n)-x^i(n)\|^2]\le A\beta(n)(1+\max_{k<n}E[\|y(k)-x^i(k)\|^2])$$

and since $\beta(n)$ converges to zero, it follows that $E[\|y(n)-x^i(n)\|^2]$ converges to zero as well. We also conclude from (A.35) that $\sup_n E[b^2(n)] < \infty$.

Let

$$D_k=\sum_{k^{1/\delta}\le i<(k+1)^{1/\delta}}\frac{1}{i}b(i), \qquad k\ge \qquad (A.39)$$

Using the fact that there exists an A such that $(k+1)^{1/\delta}-k^{1/\delta}\le Ak^{(1/\delta)-1}$, $\forall k$, we obtain from (A.39)

$$\sum_{k=1}^{\infty}E[D_k^2]\le\sum_{k=1}^{\infty}\left[\sum_{k^{1/\delta}\le i<(k+1)^{1/\delta}}\frac{1}{i}\right]^2\sup_n E[b^2(n)]$$

$$\le A\sum_{k=1}^{\infty}\frac{1}{k^{2/\delta}}[(k+1)^{1/\delta}-k^{1/\delta}]^2$$

$$\le A\sum_{k=1}^{\infty}\frac{1}{k^{2/\delta}}k^{(2/\delta)-2}<\infty. \qquad (A.40)$$

It follows that D_k^2 converges to zero, almost surely. Consequently, so does D_k and $\sum_{k=1}^{n}d^{n-k}D_k$ as well. Let us fix some n, let N denote the largest integer such that $N\le n^\delta$ and use (A.36) to obtain

$$\|x^i(n)-y(n)\|\le A\sum_{k=1}^{n-1}\frac{1}{k}c(n|k)b(k)$$

$$\le A\sum_{1\le k\le n^\delta-1}\sum_{k^{1/\delta}\le i<(k+1)^{1/\delta}}\left[\frac{1}{i}d^{n^\delta-i^\delta}b(i)\right]$$

$$\le A\sum_{1\le k\le n^\delta-}d^{N-(k+1)}\sum_{k^{1/\delta}\le i<(k+1)^{1/\delta}}\frac{1}{i}b(i)$$

$$\le A\sum_{k=1}^{N}d^{N-k}D_k. \qquad (A.41)$$

As n converges to infinity, so does N and, by the above discussion, $x^i(n) - y(n)$ converges to zero, as $n \to \infty$. Consequently, $x^i(n) - x^j(n)$ also converges to zero, for any i, j, completing the proof of part b).

Finally, since $J(y(n))$ converges and $x^i(n) - y(n)$ converges to zero, part a) of the theorem follows, as in the proof of Theorem 3.1. ■

Proof of Corollary 3.1: From part c) of either Theorem 3.1 or 3.2 and (3.7) we obtain

$$\sum_{i=1}^{M}\sum_{l=1}^{L}\sum_{n\in T_l^i}\gamma^i(n)g_l^i(x^i(n))<\infty, \qquad \text{a.s.} \qquad (A.42)$$

Because of our assumption on the sets T_l^i, it follows that if $T_l^i\ne\phi$, then there exists a positive integer c such that, for any i, l, m, the interval $\{cm+1, cm+2, \cdots, c(m+1)\}$ contains at least one element of T_l^i. Let us choose sequences of such elements denoted by $t_{l,m}^i$. By (A.42), we have

$$\sum_{i=1}^{M}\sum_{l=1}^{L}\sum_{m=1}^{\infty}\gamma^i(t_{l,m}^i)g_l^i(x^i(t_{l,m}^i))<\infty, \qquad \text{a.s.} \qquad (A.43)$$

Now notice that, for some constant $K_5 > 0$,

$$\gamma^i(t_{l,m}^i)\ge\frac{K_4}{t_{l,m}^i}\ge\frac{K_4}{c(m+1)}\ge\frac{K_5}{m}, \qquad \forall\ i,\ l,\ m.$$

Hence, (A.43) yields

$$\sum_{m=1}^{\infty}\frac{1}{m}\sum_{i=1}^{M}\sum_{l=1}^{L}g_l^i(x^i(t_{l,m}^i))<\infty, \qquad \text{a.s.} \qquad (A.45)$$

Let us assume, without any loss of generality that $T_1^1\ne\phi$. From either Theorem 3.1 or 3.2 and its proof we obtain $\lim_{n\to\infty}(x^i(n)-y(n))=\lim_{n\to\infty}(y(n+1)-y(n))=0$ which implies that

$$\lim_{m\to\infty}(x^i(t_{l,m}^i)-y(t_{1,m}^1))=0, \qquad \forall\ i,\ l. \qquad (A.46)$$

Since J has compact level sets and $J(y(n))$ converges, the sequence $\{y(n)\}$ is bounded. We therefore need to consider the functions g_l^i only on a compact set on which they are uniformly continuous. Therefore,

$$\lim_{m\to\infty}(g_l^i(x^i(t_{l,m}^i))-g_l^i(y(t_{1,m}^1)))=0, \qquad \forall\ i,\ l.$$

By combining (A.45) and (A.47) we obtain

$$\liminf_{m\to\infty}\sum_{i=1}^{M}\sum_{l=1}^{L}g_l^i(y(t_{1,m}^1))=0.$$

a) By (A.48), there must be some subsequence of $\{t_{1,m}^1\}$ along which $g(y(t_{1,m}^1))$ converges to zero. Let y^* be a limit point of the corresponding subsequence of $\{y(t_{1,m}^1)\}$. By continuity, $g(y^*)=0$ and, by assumption, y^* must be a stationary point of J, so $\nabla J(y^*)=0$. Moreover, $x^i(t_{1,m}^1)$ also converges to y^* along the same subsequence. By continuity of ∇J, (3.8) follows.

b) In this case, (A.43) implies

$$\lim_{m\to\infty}\sum_{i=1}^{M}\sum_{l=1}^{L}g_l^i(x^i(t_{l,m}^i))=0, \qquad \text{a.s.} \qquad (A.49)$$

and the rest of the proof is the same as for part a), except that we do not need to restrict ourselves to a convergent subsequence.

c) From part a) we conclude that some subsequence of $\{y(t_{1,m}^1)\}$ converges to some y^* for which $g(y^*)=0$. Consequently, y^* minimizes J. Using the continuity of J,

$$\liminf_{n\to\infty}J(y(n))\le\liminf_{n\to\infty}J(y(t_{1,n}^1))\le J(y^*)=\inf_{x\in H}J(x).$$

On the other hand, $J(y(n))$ converges (part a) of either Theorem 3.1 or 3.2) which shows that (3.10) holds. ■

References

[1] K. J. Arrow and L. Hurwicz, "Decentralization and computation in resource allocation," in *Essays in Economics and Econometrics*, R. W. Pfouts, Ed. Chapel Hill, NC: Univ. North Carolina Press, 1960, pp. 34-104.

[2] M. Avriel, *Nonlinear Programming*. Englewood Cliffs, NJ: Prentice-Hall, 1976.

[3] G. M. Baudet, "Asynchronous iterative methods for multiprocessors," *J. ACM*, vol. 25, no. 2, pp. 226-244, 1978.

[4] D. P. Bertsekas, "Distributed dynamic programming," *IEEE Trans. Automat. Contr.*, vol. AC-27, no. 3, pp. 610-616, 1982.

[5] ——, "Distributed asynchronous computation of fixed points," *Math. Programm.*, vol. 27, pp. 107-120, 1983.

[6] ——, "Optimal routing and flow control methods for communication networks," in *Analysis and Optimization of Systems*, A. Bensoussan and J. L. Lions, Eds. New York: Springer-Verlag, 1982, pp. 615-643.

[7] D. Chazan and W. Miranker, "Chaotic relaxation," *Linear Algebra and Appl.*, vol. 2, pp. 199-222, 1969.

[8] R. G. Gallager, "A minimum delay routing algorithm using distributed computation," *IEEE Trans. Commun.*, vol. COM-25, no. 1, pp. 73-85, 1977.

[9] H. T. Kung, "Synchronized and asynchronous parallel algorithms for multiprocessors," in *Algorithms and Complexity*. New York: Academic, pp. 153-200, 1976.

[10] H. J. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems* (Applied Math. Series, No. 26). New York: Springer-Verlag, 1978.

[11] L. Ljung, "Analysis of recursive stochastic algorithms," *IEEE Trans. Automat. Contr.*, vol. AC-22, pp. 551-575, 1977.

[12] L. Ljung and T. Soderstrom, *Theory and Practice of Recursive Identification*. Cambridge, MA: M.I.T. Press, 1983.

[13] P. A. Meyer, *Probability and Potentials*. Waltham, MA: Blaisdell, 1966.

[14] B. T. Poljak and Y. Z. Tsypkin, "Pseudogradient adaptation and training algorithms," *Automat. Remote Contr.*, no. 3, pp. 45-68, 1973.

[15] E. Seneta, *Non-Negative Matrices and Markov Chains*. New York: Springer-Verlag, 1981.

[16] V. Solo, "The convergence of AML," *IEEE Trans. Automat. Contr.*, vol. AC-24, pp. 958-962, 1979.

[17] J. N. Tsitsiklis and D. P. Bertsekas, "Distributed asynchronous optimal routing for data networks," in *Proc. 23d Conf. Decision Contr.*, Las Vegas, NV, Dec. 1984; also in *IEEE Trans. Automat. Contr.*, vol. AC-31, pp. 325-332, 1986.

[18] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, Dep. Elect. Eng. Comput. Sci., M.I.T., Cambridge, MA, 1984.

[19] D. P. Bertsekas and R. G. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1986.

John N. Tsitsiklis (S'80-M'81), for a photograph and biography, see p. 332 of the April 1986 issue of this TRANSACTIONS.

Dimitri P. Bertsekas (S'70-SM'77-F'84), for a photograph and biography, see p. 332 of the April 1986 issue of this TRANSACTIONS.

Michael Athans (S'58-M'61-SM'69-F'73) received the Ph.D. degree in electrical engineering in 1961 from the University of California, Berkeley.

From 1961 to 1964 he was with the M.I.T. Lincoln Laboratory, Lexington. Since 1964 he has been a faculty member in the Department of Electrical Engineering and Computer Science, M.I.T., Cambridge, where he currently holds the rank of Professor of Systems Science and Engineering. While at M.I.T., he also served as Director of the Laboratory for Information and Decision Systems from 1974 to 1981. He is a cofounder of ALPHATECH, Inc., Burlington, MA. He has also consulted for several other industrial organizations and government panels.