Gossip Dual Averaging for Decentralized Optimization of Pairwise Functions

Igor Colin IGOR.COLIN@TELECOM-PARISTECH.FR
LTCI, CNRS, Télécom ParisTech, Unversité Paris-Saclay, 75013 Paris, France

Aurélien Bellet AURELIEN.BELLET@INRIA.FR
Magnet Team, INRIA Lille – Nord Europe, 59650 Villeneuve d'Ascq, France

Joseph Salmon JOSEPH.SALMON@TELECOM-PARISTECH.FR
Stéphan Clémençon STEPHAN.CLEMENCON@TELECOM-PARISTECH.FR
LTCI, CNRS, Télécom ParisTech, Unversité Paris-Saclay, 75013 Paris, France

Abstract

In decentralized networks (of sensors, connected objects, *etc.*), there is an important need for efficient algorithms to optimize a global cost function, for instance to learn a global model from the local data collected by each computing unit. In this paper, we address the problem of decentralized minimization of pairwise functions of the data points, where these points are distributed over the nodes of a graph defining the communication topology of the network. This general problem finds applications in ranking, distance metric learning and graph inference, among others. We propose new gossip algorithms based on dual averaging which aims at solving such problems both in synchronous and asynchronous settings. The proposed framework is flexible enough to deal with constrained and regularized variants of the optimization problem. Our theoretical analysis reveals that the proposed algorithms preserve the convergence rate of centralized dual averaging up to an additive bias term. We present numerical simulations on Area Under the ROC Curve (AUC) maximization and metric learning problems which illustrate the practical interest of our approach.

1. Introduction

The increasing popularity of large-scale and fully decentralized computational architectures, fueled for instance by the advent of the "Internet of Things", motivates the development of efficient optimization algorithms adapted to this setting. An important application is machine learning in wired and wireless networks of agents (sensors, connected objects, mobile phones, *etc.*), where the agents seek to minimize a global learning objective which depends of the data collected locally by each agent. In such networks, it is typically impossible to efficiently centralize data or to globally aggregate intermediate results: agents can only communicate with their immediate neighbors (*e.g.,* agents within a small distance), often in a completely asynchronous fashion. Standard distributed optimization and machine learning algorithms (implemented for instance using MapReduce/Spark) require a coordinator node and/or to maintain synchrony, and are thus unsuitable for use in decentralized networks.

In contrast, *gossip algorithms* (Tsitsiklis, 1984; Boyd et al., 2006; Kempe et al., 2003; Shah, 2009) are tailored to this setting because they only rely on simple peer-to-peer communication: each agent only exchanges information with one neighbor at a time. Various gossip algorithms have been proposed to solve the flagship problem of decentralized optimization, namely to find a parameter vector θ which minimizes an average of convex functions $(1/n)\sum_{i=1}^{n} f(\theta; x_i)$, where the data x_i is only known to agent i. The most popular algorithms are based on (sub)gradient descent (Johansson et al., 2010; Nedić & Ozdaglar, 2009; Ram et al., 2010; Bianchi & Jakubowicz, 2013), ADMM (Wei & Ozdaglar, 2012; 2013; Iutzeler et al., 2013) or dual averaging (Duchi et al., 2012; Yuan et al., 2012; Lee et al., 2015; Tsianos et al., 2015), some of which can also accommodate constraints or regularization on θ. The main idea underlying these methods is that each agent seeks to minimize its local function by applying local updates (*e.g.,* gradient steps) while exchanging infor-

Proceedings of the 33^{rd} International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48.

mation with neighbors to ensure a global convergence to the consensus value.

In this paper, we tackle the problem of minimizing an average of *pairwise* functions of the agents' data:

$$\min_{\theta} \frac{1}{n^2} \sum_{1 \leq i,j \leq n} f(\theta; x_i, x_j). \tag{1}$$

This problem finds numerous applications in statistics and machine learning, *e.g.,* Area Under the ROC Curve (AUC) maximization (Zhao et al., 2011), distance/similarity learning (Bellet et al., 2015), ranking (Clémençon et al., 2008), supervised graph inference (Biau & Bleakley, 2006) and multiple kernel learning (Kumar et al., 2012), to name a few. As a motivating example, consider a mobile phone application which locally collects information about its users. The provider could be interested in learning pairwise similarity functions between users in order to group them into clusters or to recommend them content without having to centralize data on a server (which would be costly for the users' bandwidth) or to synchronize phones.

The main difficulty in Problem (1) comes from the fact that each term of the sum depends on two agents i and j, making the local update schemes of previous approaches impossible to apply unless data is exchanged between nodes. Although gossip algorithms have recently been introduced to evaluate such pairwise functions for a *fixed* θ (Pelckmans & Suykens, 2009; Colin et al., 2015), to the best of our knowledge, efficiently finding the *optimal solution* θ in a decentralized way remains an open challenge. Our contributions towards this objective are as follows. We propose new gossip algorithms based on dual averaging (Nesterov, 2009; Xiao, 2010) to efficiently solve Problem (1) and its constrained or regularized variants. Central to our methods is a light data propagation scheme which allows the nodes to compute *biased* estimates of the gradients of functions in (1). We then propose a theoretical analysis of our algorithms both in synchronous and asynchronous settings establishing their convergence under an additional hypothesis that the bias term decreases fast enough over the iterations (and we have observed such a fast decrease in all our experiments). Finally, we present some numerical simulations on Area Under the ROC Curve (AUC) maximization and metric learning problems. These experiments illustrate the practical performance of the proposed algorithms and the influence of network topology, and show that in practice the influence of the bias term is negligible as it decreases very fast with the number of iterations.

The paper is organized as follows. Section 2 formally introduces the problem of interest and briefly reviews the dual averaging method, which is at the root of our approach. Section 3 presents the proposed gossip algorithms and their convergence analysis. Section 4 displays our numerical simulations. Finally, concluding remarks are collected in Section 5.

2. Preliminaries

2.1. Definitions and Notation

For any integer $p > 0$, we denote by $[p]$ the set $\{1, \dots, p\}$ and by $|F|$ the cardinality of any finite set F. We denote an undirected graph by $\mathcal{G} = (V, E)$, where $V = [n]$ is the set of vertices and $E \subseteq V \times V$ is the set of edges. A node $i \in V$ has degree $d_i = |\{j : (i,j) \in E\}|$. $\mathcal{G}$ is connected if for all $(i,j) \in V^2$ there exists a path connecting i and j; it is bipartite if there exist $S, T \subset V$ such that $S \cup T = V$, $S \cap T = \emptyset$ and $E \subseteq (S \times T) \cup (T \times S)$. The graph Laplacian of $\mathcal{G}$ is denoted by $L(\mathcal{G}) = D(\mathcal{G}) - A(\mathcal{G})$, where $D(\mathcal{G})$ and $A(\mathcal{G})$ are respectively the degree and the adjacency matrices of $\mathcal{G}$.

The transpose of a matrix $M \in \mathbb{R}^{n \times n}$ is denoted by $M^\top$. A matrix $P \in \mathbb{R}^{n \times n}$ is termed stochastic whenever $P \geq 0$ and $P\mathbf{1}_n = \mathbf{1}_n$, where $\mathbf{1}_n = (1, \dots, 1)^\top \in \mathbb{R}^n$, and bistochastic whenever both P and $P^\top$ are stochastic. We denote by I_n the identity matrix in $\mathbb{R}^{n \times n}$, by $(e_1, \dots, e_n)$ the canonical basis of $\mathbb{R}^n$, by $\mathbb{I}_{\{\mathcal{E}\}}$ the indicator function of any event $\mathcal{E}$ and by $\|\cdot\|$ the usual ℓ_2-norm. For $\theta \in \mathbb{R}^d$ and $g : \mathbb{R}^d \to \mathbb{R}$, we denote by $\nabla g(\theta)$ the gradient of g at θ. Finally, given a collection of vectors $u_1, \dots, u_n$, we denote by $\bar{u}^n = (1/n) \sum_{i=1}^n u_i$ its empirical mean.

2.2. Problem Statement

We represent a network of n agents as an undirected graph $\mathcal{G} = ([n], E)$, where each node $i \in [n]$ corresponds to an agent and $(i,j) \in E$ if nodes i and j can exchange information directly (*i.e.*, they are neighbors). For ease of exposition, we assume that each node $i \in [n]$ holds a single data point $x_i \in \mathcal{X}$. Though restrictive in practice, this assumption can easily be relaxed, but it would lead to more technical details to handle the storage size, without changing the overall analysis (see supplementary material for details).

Given $d > 0$, let $f : \mathbb{R}^d \times \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a differentiable and convex function with respect to the first variable. We assume that for any $(x, x') \in \mathcal{X}^2$, there exists $L_f > 0$ such that $f(\cdot; x, x')$ is L_f-Lipschitz (with respect to the ℓ_2-norm). Let $\psi : \mathbb{R}^d \to \mathbb{R}^+$ be a non-negative, convex, possibly non-smooth, function such that, for simplicity, $\psi(0) = 0$. We aim at solving the following optimization problem:

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n^2} \sum_{1 \leq i,j \leq n} f(\theta; x_i, x_j) + \psi(\theta). \tag{2}$$

In a typical machine learning scenario, Problem (2) is a (regularized) empirical risk minimization problem and θ

Algorithm 1 Stochastic dual averaging in the centralized setting

Require: Step size $(\gamma(t))_{t\geq 0} > 0$.
1: Initialization: $\theta = 0$, $\bar{\theta} = 0$, $z = 0$.
2: **for** $t = 1, \dots, T$ **do**
3: Update $z \leftarrow z + g(t)$, where $\mathbb{E}[g(t)|\theta] = \nabla \bar{f}^n(\theta)$
4: Update $\theta \leftarrow \pi_t(z)$
5: Update $\bar{\theta} \leftarrow \left(1 - \frac{1}{t}\right)\bar{\theta} + \frac{1}{t}\theta$
6: **end for**
7: **return** $\bar{\theta}$

corresponds to the model parameters to be learned. The quantity $f(\theta; x_i, x_j)$ is a pairwise loss measuring the performance of the model θ on the data pair (x_i, x_j), while $\psi(\theta)$ represents a regularization term penalizing the complexity of θ. Common examples of regularization terms include indicator functions of a closed convex set to model explicit convex constraints, or norms enforcing specific properties such as sparsity (a canonical example being the ℓ_1-norm).

Many machine learning problems can be cast as Problem (2). For instance, in AUC maximization (Zhao et al., 2011), binary labels $(\ell_1, \dots, \ell_n) \in \{-1, 1\}^n$ are assigned to the data points and we want to learn a (linear) scoring rule $x \mapsto x^\top \theta$ which hopefully gives larger scores to positive data points than to negative ones. One may use the logistic loss

$$f(\theta; x_i, x_j) = \mathbb{I}_{\{\ell_i > \ell_j\}} \log\left(1 + \exp((x_j - x_i)^\top \theta)\right),$$

and the regularization term $\psi(\theta)$ can be the square ℓ_2-norm of θ (or the ℓ_1-norm when a sparse model is desired). Other popular instances of Problem (2) include metric learning (Bellet et al., 2015), ranking (Clémençon et al., 2008), supervised graph inference (Biau & Bleakley, 2006) and multiple kernel learning (Kumar et al., 2012).

For notational convenience, we denote by f_i the partial function $(1/n)\sum_{j=1}^n f(\cdot; x_i, x_j)$ for $i \in [n]$ and by $\bar{f}^n = (1/n)\sum_{i=1}^n f_i$. Problem (2) can then be recast as:

$$\min_{\theta \in \mathbb{R}^d} R_n(\theta) = \bar{f}^n(\theta) + \psi(\theta). \tag{3}$$

Note that the function $\bar{f}^n$ is L_f-Lipschitz, since all the f_i are L_f-Lipschitz.

Remark 1. Throughout the paper we assume that the function f is differentiable, but we expect all our results to hold even when f is non-smooth, for instance in L_1-regression problems or when using the hinge loss. In this case, one simply needs to replace gradients by subgradients in our algorithms, and a similar analysis could be performed.

2.3. Centralized Dual Averaging

In this section, we review the stochastic dual averaging optimization algorithm (Nesterov, 2009; Xiao, 2010) to solve Problem (2) in the centralized setting (where all data lie on the same machine). This method is at the root of our gossip algorithms, for reasons that will be made clear in Section 3. To explain the main idea behind dual averaging, let us first consider the iterations of Stochastic Gradient Descent (SGD), assuming $\psi \equiv 0$ for simplicity:

$$\theta(t+1) = \theta(t) - \gamma(t) g(t),$$

where $\mathbb{E}[g(t)|\theta(t)] = \nabla \bar{f}^n(\theta(t))$, and $(\gamma(t))_{t\geq 0}$ is a non-negative non-increasing step size sequence. For SGD to converge to an optimal solution, the step size sequence must satisfy $\gamma(t) \underset{t\to+\infty}{\longrightarrow} 0$ and $\sum_{t=0}^{\infty} \gamma(t) = \infty$. As noticed by Nesterov (2009), an undesirable consequence is that new gradient estimates are given smaller weights than old ones. Dual averaging aims at integrating all gradient estimates with the same weight.

Let $(\gamma(t))_{t\geq 0}$ be a positive and non-increasing step size sequence. The dual averaging algorithm maintains a sequence of iterates $(\theta(t))_{t>0}$, and a sequence $(z(t))_{t\geq 0}$ of "dual" variables which collects the sum of the unbiased gradient estimates seen up to time t. We initialize to $\theta(1) = z(0) = 0$. At each step $t > 0$, we compute an unbiased estimate $g(t)$ of $\nabla \bar{f}^n(\theta(t))$. The most common choice is to take $g(t) = \nabla f(\theta; x_{i_t}, x_{j_t})$ where i_t and j_t are drawn uniformly at random from $[n]$. We then set $z(t+1) = z(t) + g(t)$ and generate the next iterate with the following rule:

$$\begin{cases} \theta(t+1) = \pi_t^\psi(z(t+1)), \\ \pi_t^\psi(z) := \underset{\theta \in \mathbb{R}^d}{\arg\min} \left\{ -z^\top \theta + \dfrac{\|\theta\|^2}{2\gamma(t)} + t\psi(\theta) \right\}. \end{cases}$$

When it is clear from the context, we will drop the dependence in ψ and simply write $\pi_t(z) = \pi_t^\psi(z)$.

Remark 2. Note that $\pi_t(\cdot)$ is related to the proximal operator of a function $\phi : \mathbb{R}^d \to \mathbb{R}$ defined by $\mathrm{prox}_\phi(x) = \arg\min_{z\in\mathbb{R}^d} \left(\|z - x\|^2/2 + \phi(x)\right)$. Indeed, one can write:

$$\pi_t(z) = \mathrm{prox}_{t\gamma(t)\psi}\left(\gamma(t) z\right).$$

For many functions ψ of practical interest, $\pi_t(\cdot)$ has a closed form solution. For instance, when $\psi = \|\cdot\|^2$, $\pi_t(\cdot)$ corresponds to a simple scaling, and when $\psi = \|\cdot\|_1$ it is a soft-thresholding operator. If ψ is the indicator function of a closed convex set $\mathcal{C}$, then $\pi_t(\cdot)$ is the projection operator onto $\mathcal{C}$.

The dual averaging method is summarized in Algorithm 1. If $\gamma(t) \propto 1/\sqrt{t}$ then for any $T > 0$:

$$\mathbb{E}_T\left[R_n(\bar{\theta}(T)) - R_n(\theta^*)\right] = \mathcal{O}(1/\sqrt{T}),$$

where $\theta^* \in \arg\min_{\theta\in\mathbb{R}^d} R_n(\theta)$, $\bar{\theta}(T) = \frac{1}{T}\sum_{i=1}^T \theta(t)$ is the averaged iterate and $\mathbb{E}_T$ is the expectation over all possible sequences $(g(t))_{1\leq t\leq T}$. A precise statement of this

result along with a proof can be found in the supplementary material for completeness.

Notice that dual averaging cannot be easily adapted to our decentralized setting. Indeed, a node cannot compute an unbiased estimate of its gradient: this would imply an access to the entire set of data points, which violates the communication and storage constraints. Therefore, data points have to be appropriately propagated during the optimization procedure, as detailed in the following section.

3. Pairwise Gossip Dual Averaging

We now turn to our main goal, namely to develop efficient gossip algorithms for solving Problem (2) in the decentralized setting. The methods we propose rely on dual averaging (see Section 2.3). This choice is guided by the fact that the structure of the updates makes dual averaging much easier to analyze in the distributed setting than sub-gradient descent when the problem is constrained or regularized. This is because dual averaging maintains a simple sum of sub-gradients, while the (non-linear) smoothing operator π_t is applied separately.

Our work builds upon the analysis of Duchi et al. (2012), who proposed a distributed dual averaging algorithm to optimize an average of *univariate* functions $f(\cdot; x_i)$. In their algorithm, each node i computes *unbiased* estimates of its local function $\nabla f(\cdot; x_i)$ that are iteratively averaged over the network. Unfortunately, in our setting, the node i cannot compute unbiased estimates of $\nabla f_i(\cdot) = \nabla(1/n)\sum_{j=1}^n f(\cdot; x_i, x_j)$: the latter depends on all data points while each node $i \in [n]$ only holds x_i. To go around this problem, we rely on a gossip data propagation step (Pelckmans & Suykens, 2009; Colin et al., 2015) so that the nodes are able to compute *biased* estimates of $\nabla f_i(\cdot)$ while keeping the communication and memory overhead to a small level for each node.

We present and analyze our algorithm in the synchronous setting in Section 3.1. We then turn to the more intricate analysis of the asynchronous setting in Section 3.2.

3.1. Synchronous Setting

In the synchronous setting, we assume that each node has access to a global clock such that every node can update simultaneously at each tick of the clock. Although not very realistic, this setting allows for simpler analysis. We assume that the scaling sequence $(\gamma(t))_{t\geq 0}$ is the same for every node. At any time, each node i has the following quantities in its local memory register: a variable z_i (the gradient accumulator), its original observation x_i, and an *auxiliary observation* y_i, which is initialized at x_i but will change throughout the algorithm as a result of data propagation.

Algorithm 2 Gossip dual averaging for pairwise function in synchronous setting

Require: Step size $(\gamma(t))_{t\geq 1} > 0$.
Each node i initializes $y_i = x_i$, $z_i = \theta_i = \bar{\theta}_i = 0$.
for $t = 1, \dots, T$ **do**
  Draw (i, j) uniformly at random from E
  Set $z_i, z_j \leftarrow \frac{z_i + z_j}{2}$
  Swap auxiliary observations: $y_i \leftrightarrow y_j$
  for $k = 1, \dots, n$ **do**
    Update $z_k \leftarrow z_k + \nabla_\theta f(\theta_k; x_k, y_k)$
    Compute $\theta_k \leftarrow \pi_t(z_k)$
    Average $\bar{\theta}_k \leftarrow \left(1 - \frac{1}{t}\right)\bar{\theta}_k + \frac{1}{t}\theta_k$
  end for
end for
return Each node k has $\bar{\theta}_k$

The algorithm goes as follows. At each iteration, an edge $(i, j) \in E$ of the graph is drawn uniformly at random. Then, nodes i and j average their gradient accumulators z_i and z_j, and swap their auxiliary observations y_i and y_j. Finally, every node of the network performs a dual averaging step, using their original observation and their current auxiliary one to estimate the partial gradient. The procedure is detailed in Algorithm 2, and the following proposition adapts the convergence rate of centralized dual averaging under the hypothesis that the contribution of the bias term decreases fast enough over the iterations.

Theorem 1. *Let $\mathcal{G}$ be a connected and non-bipartite graph with n nodes, and let $\theta^* \in \arg\min_{\theta\in\mathbb{R}^d} R_n(\theta)$. Let $(\gamma(t))_{t\geq 1}$ be a non-increasing and non-negative sequence. For any $i \in [n]$ and any $t \geq 0$, let $z_i(t) \in \mathbb{R}^d$ and $\bar{\theta}_i(t) \in \mathbb{R}^d$ be generated according to Algorithm 2. Then for any $i \in [n]$ and $T > 1$, we have:*

$$\mathbb{E}_T[R_n(\bar{\theta}_i) - R_n(\theta^*)] \leq C_1(T) + C_2(T) + C_3(T),$$

where

$$\begin{cases} C_1(T) = \dfrac{1}{2T\gamma(T)}\|\theta^*\|^2 + \dfrac{L_f^2}{2T}\displaystyle\sum_{t=1}^{T-1}\gamma(t), \\ C_2(T) = \dfrac{3L_f^2}{T\left(1 - \sqrt{\lambda_2^{\mathcal{G}}}\right)}\displaystyle\sum_{t=1}^{T-1}\gamma(t), \\ C_3(T) = \dfrac{1}{T}\displaystyle\sum_{t=1}^{T-1}\mathbb{E}_t[(\omega(t) - \theta^*)^\top \bar{\epsilon}^n(t)], \end{cases}$$

and $\lambda_2^{\mathcal{G}} < 1$ is the second largest eigenvalue of the matrix $W(\mathcal{G}) = I_n - \frac{1}{|E|}L(\mathcal{G})$.

Sketch of proof. First notice that at a given (outer) iteration $t + 1$, $\bar{z}^n$ is updated as follows:

$$\bar{z}^n(t+1) = \bar{z}^n(t) + \frac{1}{n}\sum_{k=1}^n d_k(t), \tag{4}$$

where $d_k(t) = \nabla_\theta f(\theta_k(t); x_k, y_k(t+1))$ is a biased estimate of $\nabla f_k(\theta_k(t))$. Let $\epsilon_k(t) = d_k(t) - g_k(t)$ be the bias, so that we have $\mathbb{E}[g_k(t)|\theta_k(t)] = \nabla f_k(\theta_k(t))$.

Let us define $\omega(t) = \pi_t(\bar{z}^n(t))$. Using convexity of R_n, the gradient's definition and the fact that the functions $\bar{f}^n$ and π_t are both L_f-Lipschitz, we obtain: for $T \geq 2$ and $i \in [n]$,

$$\begin{aligned}
&\mathbb{E}_T[R_n(\bar{\theta}_i(T)) - R_n(\theta^*)] \\
&\quad\leq \frac{L_f}{nT}\sum_{t=2}^{T}\gamma(t-1)\sum_{j=1}^{n}\mathbb{E}_t\Big[\|z_i(t) - z_j(t)\|\Big] \quad (5)\\
&\quad+ \frac{L_f}{nT}\sum_{t=2}^{T}\gamma(t-1)\sum_{j=1}^{n}\mathbb{E}_t\Big[\|\bar{z}^n(t) - z_j(t)\|\Big] \quad (6)\\
&\quad+ \frac{1}{T}\sum_{t=2}^{T}\mathbb{E}_t[(\omega(t)-\theta^*)^\top \bar{g}^n(t)]. \quad (7)
\end{aligned}$$

Using Lemma 4 (see supplementary material), the terms (5)-(6) can be bounded by $C_2(T)$. The term (7) requires a specific analysis because the updates are performed using biased estimates. We decompose it as follows:

$$\begin{aligned}
&\frac{1}{T}\sum_{t=2}^{T}\mathbb{E}_t\Big[\omega(t)-\theta^*)^\top \bar{g}^n(t)\Big] \\
&\quad= \frac{1}{T}\sum_{t=2}^{T}\mathbb{E}_t\Big[(\omega(t)-\theta^*)^\top(\bar{d}^n(t) - \bar{\epsilon}^n(t))\Big] \\
&\quad\leq \frac{1}{T}\sum_{t=2}^{T}\mathbb{E}_t\Big[(\omega(t)-\theta^*)^\top \bar{d}^n(t)\Big] \quad (8)\\
&\quad+ \frac{1}{T}\sum_{t=2}^{T}\mathbb{E}_t\Big[(\omega(t)-\theta^*)^\top \bar{\epsilon}^n(t)\Big].
\end{aligned}$$

The term (8) can be bounded by $C_1(T)$ (see Xiao, 2010, Lemma 9). We refer the reader to the supplementary material for the detailed proof. □

The rate of convergence in Proposition 1 is divided into three parts: $C_1(T)$ is a *data dependent* term which corresponds to the rate of convergence of the centralized dual averaging, while $C_2(T)$ and $C_3(T)$ are *network dependent* terms since $1 - \lambda_2^{\mathcal{G}} = \beta_{n-1}^{\mathcal{G}}/|E|$, where $\beta_{n-1}^{\mathcal{G}}$ is the second smallest eigenvalue of the graph Laplacian $L(\mathcal{G})$, also known as the spectral gap of $\mathcal{G}$. The convergence rate of our algorithm thus improves when the spectral gap is large, which is typically the case for well-connected graphs (Chung, 1997). Note that $C_2(T)$ corresponds to the network dependence for the distributed dual averaging algorithm of Duchi et al. (2012) while the term $C_3(T)$ comes from the bias of our partial gradient estimates. In practice, $C_3(T)$ vanishes quickly and has a small impact on the rate of convergence, as shown in Section 4.

Algorithm 3 Gossip dual averaging for pairwise function in asynchronous setting

Require: Step size $(\gamma(t))_{t\geq 0} > 0$, probabilities $(p_k)_{k\in[n]}$.
Each node i initializes $y_i = x_i$, $z_i = \theta_i = \bar{\theta}_i = 0$, $m_i = 0$.
for $t = 1, \dots, T$ **do**
  Draw (i, j) uniformly at random from E
  Swap auxiliary observations: $y_i \leftrightarrow y_j$
  for $k \in \{i, j\}$ **do**
    Set $z_k \leftarrow \frac{z_i + z_j}{2}$
    Update $z_k \leftarrow \frac{1}{p_k}\nabla_\theta f(\theta_k; x_k, y_k)$
    Increment $m_k \leftarrow m_k + \frac{1}{p_k}$
    Compute $\theta_k \leftarrow \pi_{m_k}(z_k)$
    Average $\bar{\theta}_k \leftarrow \left(1 - \frac{1}{m_k p_k}\right)\bar{\theta}_k$
  end for
end for
return Each node k has $\bar{\theta}_k$

3.2. Asynchronous Setting

For any variant of gradient descent over a network with a decreasing step size, there is a need for a common time scale to perform the suitable decrease. In the synchronous setting, this time scale information can be shared easily among nodes by assuming the availability of a global clock. This is convenient for theoretical considerations, but is unrealistic in practical (asynchronous) scenarios. In this section, we place ourselves in a fully asynchronous setting where each node has a local clock, ticking at a Poisson rate of 1, independently from the others. This is equivalent to a global clock ticking at a rate n Poisson process which wakes up an edge of the network uniformly at random (see Boyd et al., 2006, for details on clock modeling).

With this in mind, Algorithm 2 needs to be adapted to this setting. First, one cannot perform a full dual averaging update over the network since only two nodes wake up at each iteration. Also, as mentioned earlier, each node needs to maintain an estimate of the current iteration number in order for the scaling factor γ to be consistent across the network. For $k \in [n]$, let p_k denote the probability for the node k to be picked at any iteration. If the edges are picked uniformly at random, then one has $p_k = 2d_k/|E|$. For simplicity, we focus only on this case, although our analysis holds in a more general setting.

Let us define an activation variable $(\delta_k(t))_{t\geq 1}$ such that for any $t \geq 1$,

$$\delta_k(t) = \begin{cases} 1 & \text{if node } k \text{ is picked at iteration } t, \\ 0 & \text{otherwise.} \end{cases}$$

One can immediately see that $(\delta_k(t))_{t\geq 1}$ are i.i.d. random variables, Bernoulli distributed with parameter p_k. Let us define $(m_k(t)) \geq 0$ such that $m_k(0) = 0$ and for $t \geq 0$, $m_k(t+1) = m_k(t) + \frac{\delta_k(t+1)}{p_k}$. Since $(\delta_k(t))_{t\geq 1}$ are

Bernoulli random variables, $m_k(t)$ is an unbiased estimate of the time t.

Using this estimator, we can now adapt Algorithm 2 to the fully asynchronous case, as shown in Algorithm 3. The update step slightly differs from the synchronous case: the partial gradient has a weight $1/p_k$ instead of 1 so that all partial functions asymptotically count in equal way in every gradient accumulator. In contrast, uniform weights would penalize partial gradients from low degree nodes since the probability of being drawn is proportional to the degree. This weighting scheme is essential to ensure the convergence to the global solution. The model averaging step also needs to be altered: in absence of any global clock, the weight $1/t$ cannot be used and is replaced by $1/(m_k p_k)$, where $m_k p_k$ corresponds to the average number of times that node k has been selected so far.

The following result is the analogous of Theorem 1 for the asynchronous setting.

Theorem 2. *Let $\mathcal{G}$ be a connected and non bipartite graph. Let $(\gamma(t))_{t\geq 1}$ be defined as $\gamma(t) = c/t^{1/2+\alpha}$ for some constant $c > 0$ and $\alpha \in (0, 1/2)$. For $i \in [n]$, let $(d_i(t))_{t\geq 1}$, $(g_i(t))_{t\geq 1}$, $(\epsilon_i(t))_{t\geq 1}$, $(z_i(t))_{t\geq 1}$ and $(\theta_i(t))_{t\geq 1}$ be generated as described in Algorithm 3. Then, there exists some constant $C < +\infty$ such that, for $\theta^* \in \arg\min_{\theta' \in \mathbb{R}^d} R_n(\theta')$, $i \in [n]$ and $T > 0$,*

$$R_n(\bar{\theta}_i(T)) - R_n(\theta^*) \leq C \max(T^{-\alpha/2}, T^{\alpha-1/2}) + \frac{1}{T}\sum_{t=2}^{T} \mathbb{E}_t[(\omega(t) - \theta^*)^\top \bar{\epsilon}^n(t)].$$

The proof is given in the supplementary material.

Remark 3. In the asynchronous setting, no convergence rate was known even for the distributed dual averaging algorithm of Duchi et al. (2012), which deals with the simpler problem of minimizing *univariate* functions. The arguments used to derive Theorem 2 can be adapted to derive a convergence rate (without the bias term) for an asynchronous version of their algorithm.

Remark 4. We have focused on the setting where all pairs of observations are involved in the objective. In practice, the objective may depend only on a subset of all pairs. To efficiently apply our algorithm to this case, one should take advantage of the potential structure of the subset of interest: for instance, one could attach some additional concise information to each observation so that a node can easily identify whether a pair contributes to the objective, and if not set the loss to be zero. This is essentially the case in the AUC optimization problem studied in Section 4, where pairs of similarly labeled observations do not contribute to the objective. If the subset of pairs cannot be expressed in such a compact form, then one would need to provide each node with an index list of active pairs, which could be memory-intensive when n is large.

4. Numerical Simulations

In this section, we present numerical experiments on two popular machine learning problems involving pairwise functions: Area Under the ROC Curve (AUC) maximization and metric learning. Our results show that our algorithms converge and that the bias term vanishes very quickly with the number of iterations.

To study the influence of the network topology, we perform our simulations on three types of network (see Table 1 for the corresponding spectral gap values):

- *Complete graph:* All nodes are connected to each other. It is the ideal situation in our framework, since any pair of nodes can communicate directly. In this setting, the bias of gradient estimates should be very small, as one has for any $k \in [n]$ and any $t \geq 1$, $\mathbb{E}_t[d_k(t)|\theta_k(t)] = 1/(n-1)\sum_{y' \neq y_k(t)} \nabla_\theta f(\theta_k(t); x_k, y')$. For a network size n, the complete graph achieves the highest spectral gap: $1 - \lambda_2^{\mathcal{G}} = 1/n$, see Bollobás (1998, Ch.9) or Chung (1997, Ch.1) for details.
- *Cycle graph:* This is the worst case in terms of connectivity: each node only has two neighbors. This network has a spectral gap of order $1/n^3$, and gives a lower bound in terms of convergence rate.
- *Watts-Strogatz:* This random network generation technique (Watts & Strogatz, 1998) relies on two parameters: the average degree of the network k and a rewiring probability p. In expectation, the higher the rewiring probability, the better the connectivity of the network. Here, we use $k = 5$ and $p = 0.3$ to achieve a compromise between the connectivities of the complete graph and the cycle graph.

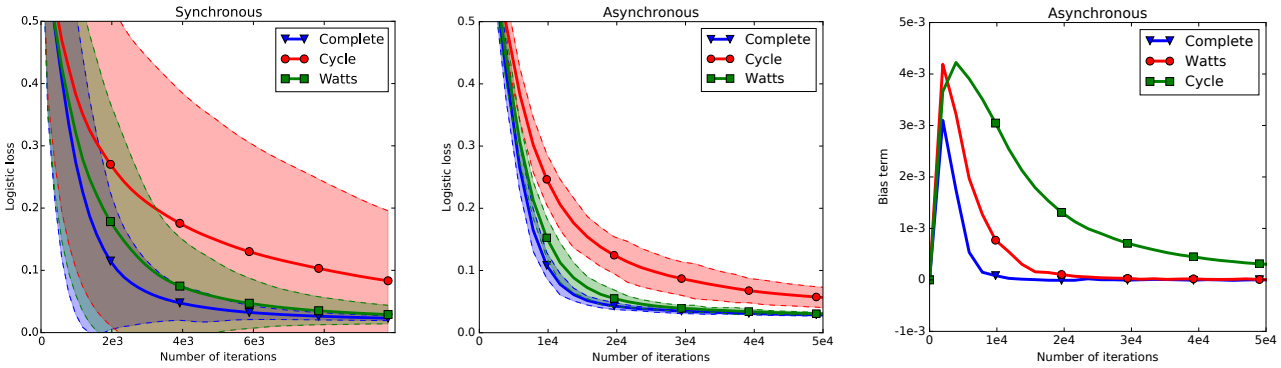
AUC Maximization We first present an application of our algorithms to AUC maximization on a real dataset. Given a set of data points $x_1, \ldots, x_n \in \mathbb{R}^d$ with associated binary labels $\ell_1, \ldots, \ell_n \in \{-1, 1\}$, the goal is to learn a linear scoring rule $x \mapsto x^\top \theta$ parameterized by $\theta \in \mathbb{R}^d$ which maximizes:

$$AUC(\theta) = \frac{\sum_{1\leq i,j\leq n} \mathbb{I}_{\{\ell_i > \ell_j\}} \mathbb{I}_{\{x_i^\top \theta > x_j^\top \theta\}}}{\sum_{1\leq i,j\leq n} \mathbb{I}_{\{\ell_i > \ell_j\}}}.$$

It corresponds to the probability that the scoring rule associated with θ outputs a higher score on a positively labeled sample than on a negatively labeled one. This formulation leads to a non-smooth optimization problem; therefore, one

Dataset	Complete graph	Watts-Strogatz	Cycle graph
Breast Cancer (AUC Maximization, $n = 699$)	$1.43 \cdot 10^{-3}$	$8.71 \cdot 10^{-5}$	$5.78 \cdot 10^{-8}$
Synthetic (Metric Learning, $n = 1000$)	$1.00 \cdot 10^{-3}$	$6.23 \cdot 10^{-5}$	$1.97 \cdot 10^{-8}$

Table 1. Spectral gap values $1 - \lambda_2^{\mathcal{G}}$ for each network.



(a) Evolution of the objective function and its standard deviation (synchronous)

(b) Evolution of the objective function and its standard deviation (asynchronous)

(c) Evolution of the bias term (asynchronous)

Figure 1. AUC maximization in synchronous and asynchronous settings.

typically minimizes a convex surrogate such as the logistic loss:

$$R_n(\theta) = \frac{1}{n^2} \sum_{1 \leq i,j \leq n} \mathbb{I}_{\{\ell_i > \ell_j\}} \log\left(1 + \exp((x_j - x_i)^\top \theta)\right).$$

We do not apply any regularization (*i.e.*, $\psi \equiv 0$), and use the Breast Cancer Wisconsin dataset,[1] which consists of $n = 699$ points in $d = 11$ dimensions.

We initialize each θ_i to 0 and for each network, we run 50 times Algorithms 2 and 3 with $\gamma(t) = 1/\sqrt{t}$.[2] Figure 1(a) shows the evolution of the objective function and the associated standard deviation (across nodes) with the number of iterations in the synchronous setting. As expected, the average convergence rate on the complete and the Watts-Strogatz networks is much better than on the poorly connected cycle network. The standard deviation of the node estimates also decreases with the connectivity of the network.

The results for the asynchronous setting are shown in Figure 1(b). As expected, the convergence rate is slower in terms of number of iterations (roughly 5 times) than in the synchronous setting. Note however that much fewer dual averaging steps are performed: for instance, on the Watts-Strogatz network, reaching a 0.1 loss requires $210,000$ (partial) gradient computations in the synchronous setting and only $25,000$ in the asynchronous setting. Moreover, the standard deviation of the estimates is much lower than in the synchronous setting. This is because communication and local optimization are better balanced in the asynchronous setting (one optimization step for each gradient accumulator averaged) than in the synchronous setting (n optimization steps for 2 gradient accumulators averaged).

The good practical convergence of our algorithm comes from the fact that the bias term $\bar{\epsilon}^n(t)^\top \omega(t)$ vanishes quite fast. Figure 1(c) shows that its average value quickly converges to 0 on all networks. Moreover, its order of magnitude is negligible compared to the objective function. In order to fully estimate the impact of this bias term on the performance, we also compare our algorithm to the ideal but unrealistic situation where each node is given an unbiased estimate of its partial gradient: instead of adding $\nabla f(\theta_i(t); x_i, y_i(t))$ to $z_i(t)$, a node i will add $\nabla f(\theta_i(t); x_i, x_j)$ where $j \in [n]$ is picked uniformly at random. As shown in Figure 2, the performance of both methods are very similar on well-connected networks.

Metric Learning We now turn to a metric learning application. We consider the family of Mahalanobis distances $D_\theta(x_i, x_j) = (x_i - x_j)^\top \theta (x_i - x_j)$ parameterized by $\theta \in \mathbb{S}_+^d$, where $\mathbb{S}_+^d$ is the cone of $d \times d$ positive semi-definite real-valued matrices. Given a set of data points $x_1, \ldots, x_n \in \mathbb{R}^d$ with associated labels $\ell_1, \ldots, \ell_n \in \{-1, 1\}$, the goal is to find $\theta \in \mathbb{S}_+^d$ which minimizes the

[1] https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original)

[2] Even if this scaling sequence does not fulfill the hypothesis of Theorem 2 for the asynchronous setting, the convergence rate is acceptable in practice.

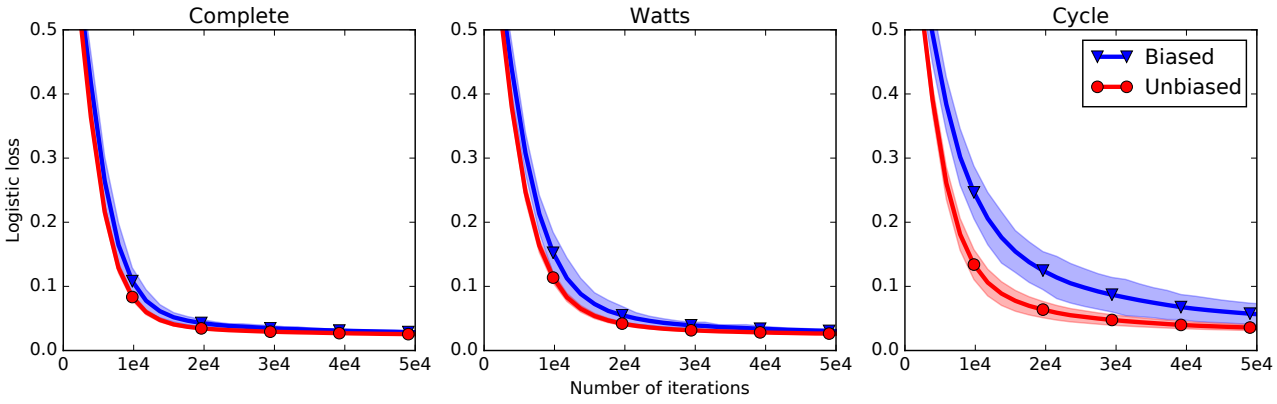
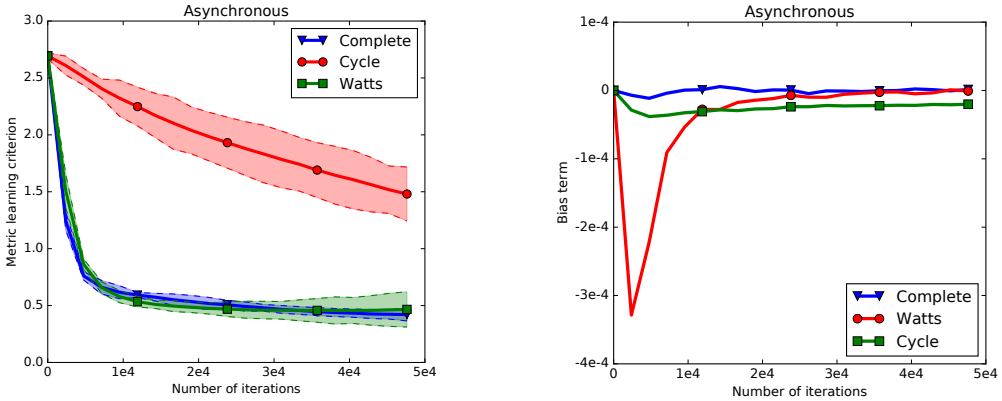


Figure 2. AUC maximization: comparison between our algorithm and an unbiased version.



(a) Evolution of the objective function and its standard deviation (asynchronous setting)

(b) Evolution of the bias term

Figure 3. Metric learning experiments.

following criterion (Jin et al., 2009):

$$R_n(\theta) = \frac{1}{n^2} \sum_{1 \leq i,j \leq n} \left[\ell_i \ell_j (b - D_\theta(x_i, x_j)) \right]_+ + \psi(\theta),$$

where $[u]_+ = \max(0, 1-u)$, $b > 0$, and $\psi(\theta) = \infty$ if $\theta \notin \mathbb{S}_+^d$ and 0 otherwise. We use a synthetic dataset of $n = 1,000$ points generated as follows: each point is drawn from a mixture of 10 Gaussians in $\mathbb{R}^{40}$ (each corresponding to a class) with all Gaussian means contained in a 5d subspace and their shared covariance matrix proportional to the identity with a variance factor such that some overlap is observed.

Figure 3(a) shows the evolution of the objective function and its standard deviation for the asynchronous setting. As in the case of AUC maximization, the algorithm converges much faster on the well-connected networks than on the cycle network. Again, we can see in Figure 3(b) that the bias vanishes very quickly with the number of iterations.

Additional Experiment We refer to the supplementary material for a metric learning experiment on a real dataset.

5. Conclusion

In this work, we have introduced new synchronous and asynchronous gossip algorithms to optimize functions depending on pairs of data points distributed over a network. The proposed methods are based on dual averaging and can readily accommodate various popular regularization terms. We provided an analysis showing that they behave similarly to the centralized dual averaging algorithm, with additional terms reflecting the network connectivity and the gradient bias. Finally, we proposed some numerical experiments on AUC maximization and metric learning which illustrate the performance of the proposed algorithms, as well as the influence of network topology. A challenging line of future research consists in designing and analyzing novel adaptive gossip schemes, where the communication scheme is dynamic and depends on the network connectivity properties and on the local information carried by each node.

Acknowledgments

This work was partially supported by the chair "Machine Learning for Big Data" of Télécom ParisTech and by a grant from CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020.

References

Bellet, Aurélien, Habrard, Amaury, and Sebban, Marc. *Metric Learning*. Morgan & Claypool, 2015.

Bianchi, Pascal and Jakubowicz, Jérémie. Convergence of a Multi-Agent Projected Stochastic Gradient Algorithm for Non-Convex Optimization. *IEEE Trans. Autom. Control*, 58(2):391–405, 2013.

Biau, Gérard and Bleakley, Kevin. Statistical Inference on Graphs. *Statistics & Decisions*, 24:209–232, 2006.

Bollobás, Béla. *Modern Graph Theory*, volume 184. Springer, 1998.

Boyd, Stephen, Ghosh, Arpita, Prabhakar, Balaji, and Shah, Devavrat. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508–2530, 2006.

Chung, Fan. *Spectral Graph Theory*, volume 92. Amer. Math. Soc., 1997.

Clémençon, Stéphan, Lugosi, Gàbor, and Vayatis, Nicolas. Ranking and Empirical Minimization of U-statistics. *Ann. Stat.*, 36(2):844–874, 2008.

Colin, I., Bellet, A., Salmon, J., and Clémençon, S. Extending Gossip Algorithms to Distributed Estimation of U-Statistics. In *NIPS*, 2015.

Duchi, John, Agarwal, Alekh, and Wainwright, Martin. Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling. *IEEE Trans. Autom. Control*, 57(3):592–606, 2012.

Iutzeler, Franck, Bianchi, Pascal, Ciblat, Philippe, and Hachem, Walid. Asynchronous Distributed Optimization using a Randomized Alternating Direction Method of Multipliers. In *IEEE CDC*, pp. 3671–3676, 2013.

Jin, R., Wang, S., and Zhou, Y. Regularized Distance Metric Learning: Theory and Algorithm. In *NIPS*, pp. 862–870, 2009.

Johansson, Björn, Rabi, Maben, and Johansson, Mikael. A Randomized Incremental Subgradient Method for Distributed Optimization in Networked Systems. *SIAM J. Optimiz.*, 20(3):1157–1170, 2010.

Kempe, David, Dobra, Alin, and Gehrke, Johannes. Gossip-Based Computation of Aggregate Information. In *FOCS*, pp. 482–491, 2003.

Kumar, Abhishek, Niculescu-Mizil, Alexandru, Kavukcuoglu, K., and Daumé, Hal. A Binary Classification Framework for Two-Stage Multiple Kernel Learning. In *ICML*, 2012.

Lee, Soomin, Nedić, Angelia, and Raginsky, Maxim. Decentralized online optimization with global objectives and local communication. *arXiv preprint arXiv:1508.07933*, 2015.

Nedić, Angelia and Ozdaglar, Asuman E. Distributed Subgradient Methods for Multi-Agent Optimization. *IEEE Trans. Autom. Control*, 54(1):48–61, 2009.

Nesterov, Yurii. Primal-dual subgradient methods for convex problems. *Math. Program.*, 120(1):261–283, 2009.

Pelckmans, Kristiaan and Suykens, Johan. Gossip Algorithms for Computing U-Statistics. In *NecSys*, pp. 48–53, 2009.

Ram, S., Nedić, Angelia, and Veeravalli, V. Distributed Stochastic Subgradient Projection Algorithms for Convex Optimization. *J. Optimiz. Theory. App.*, 147(3):516–545, 2010.

Shah, Devavrat. Gossip Algorithms. *Foundations and Trends in Networking*, 3(1):1–125, 2009.

Tsianos, Konstantinos, Lawlor, Sean, and Rabbat, Michael. Push-Sum Distributed Dual Averaging for convex optimization. In *IEEE CDC*, 2015.

Tsitsiklis, John. *Problems in decentralized decision making and computation*. PhD thesis, Massachusetts Institute of Technology, 1984.

Watts, Duncan J and Strogatz, Steven H. Collective dynamics of 'small-world'networks. *Nature*, 393(6684): 440–442, 1998.

Wei, Ermin and Ozdaglar, Asuman. Distributed Alternating Direction Method of Multipliers. In *IEEE CDC*, pp. 5445–5450, 2012.

Wei, Ermin and Ozdaglar, Asuman. On the O(1/k) Convergence of Asynchronous Distributed Alternating Direction Method of Multipliers. In *IEEE GlobalSIP*, 2013.

Xiao, Lin. Dual averaging methods for regularized stochastic learning and online optimization. *JMLR*, 11:2543–2596, 2010.

Yuan, Deming, Xu, Shengyuan, Zhao, Huanyu, and Rong, Lina. Distributed dual averaging method for multi-agent optimization with quantized communication. *Systems & Control Letters*, 61(11):1053–1061, 2012.

Zhao, Peilin, Hoi, Steven, Jin, Rong, and Yang, Tianbao. Online AUC Maximization. In *ICML*, 2011.