Journal of Computational Mathematics
Vol.35, No.4, 2017, 383–396.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1606-m2015-0452

EXTRAPUSH FOR CONVEX SMOOTH DECENTRALIZED OPTIMIZATION OVER DIRECTED NETWORKS*

Jinshan Zeng
College of Computer Information Engineering, Jiangxi Normal University, Nanchang, Jiangxi 330022, China
Email: jinshanzeng@jxnu.edu.cn
Wotao Yin
Department of Mathematics, University of California, Los Angeles, CA 90095, USA
Email: wotaoyin@math.ucla.edu

Abstract

In this note, we extend the algorithms Extra [13] and subgradient-push [10] to a new algorithm *ExtraPush* for consensus optimization with convex differentiable objective functions over a *directed* network. When the stationary distribution of the network can be computed in advance, we propose a simplified algorithm called *Normalized ExtraPush*. Just like Extra, both ExtraPush and Normalized ExtraPush can iterate with a fixed step size. But unlike Extra, they can take a column-stochastic mixing matrix, which is not necessarily doubly stochastic. Therefore, they remove the undirected-network restriction of Extra. Subgradient-push, while also works for *directed* networks, is slower on the same type of problem because it must use a sequence of diminishing step sizes.

We present preliminary analysis for ExtraPush under a bounded sequence assumption. For Normalized ExtraPush, we show that it naturally produces a bounded, linearly convergent sequence provided that the objective function is strongly convex.

In our numerical experiments, ExtraPush and Normalized ExtraPush performed similarly well. They are significantly faster than subgradient-push, even when we hand-optimize the step sizes for the latter.

Mathematics subject classification: 90C25, 90C30.
Key words: Decentralized optimization, Directed graph, Consensus, Non-doubly stochastic, Extra.

1. Introduction

We consider the following consensus optimization problem defined on a directed, strongly connected network of n agents:

$$\underset{x\in\mathbf{R}^p}{\text{minimize}}\, f(x) \triangleq \sum_{i=1}^{n} f_i(x), \tag{1.1}$$

where f_i is a proper, closed, convex, differentiable function only known to the agent i.

The model (1.1) finds applications in decentralized averaging, learning, estimation, and control. For a stationary network with *bi-directional* communication, the existing algorithms include the (sub)gradient methods [2, 5, 8, 9, 13, 19], and the primal-dual domain methods such as the decentralized alternating direction method of multipliers (DADMM) [11, 12].

* Received November 12, 2015 / Revised version received June 2, 2016 / Accepted June 27, 2016 / Published online June 1, 2017 /

This note focuses on a *directed* network (with *directional* communication), where the research of decentralized optimization is pioneered by the works [15–17]. When communication is bi-directional, algorithms can use a symmetric and doubly-stochastic mixing matrix to obtain a consensual solution; however, once the communication is directional, the mixing matrix becomes generally asymmetric and only column-stochastic. Also consider the setting where each agent broadcasts its information to its neighbors, yet an agent may not receive the information from a neighbor. An agent can weigh its information (both from itself and received from its neighbors) so that the total weights add up to 1, but an agent cannot ensure that its broadcasted information receives weights that precisely add up to exactly 1. Therefore, only each column of the mixing matrix sums to 1. In the column-stochastic setting, the push-sum protocol [6] can be used to obtain a stationary distribution for the mixing matrix.

In the symmetric and doubly-stochastic setting, if the objective is Lipschitz-differentiable, the gradient-based algorithm Extra [13] converges at the rate of $O(1/t)$, where t is the iteration number. In the column-stochastic setting, the best rate is $O(\ln t/\sqrt{t})$ from the subgradient-based algorithm [10]. We address the open question of how to take advantage of the gradient of a Lipschitz-differentiable objective. We make an attempt in this note to combine ideas in [10,13] and present our preliminary results.

Specifically, we propose *ExtraPush*, which is a two-step iteration like Extra and incorporates the push-sum protocol. At each iteration, the Extra variables are approximately normalized by the current push-sum variables. When the stationary distribution of the network can be easily computed, we propose to first apply the push-sum protocol to obtain the stationary distribution and then run the two-step iteration Normalized ExtraPush. At each iteration, its running variables are normalized by the stationary distribution.

Our algorithms are essentially the same as found in the recent work by Xi and Khan [18]. They attempted to prove convergence for a strongly convex objective function. They noticed that a certain matrix that is important to the analysis (as a part of their convergence metric) is positive semi-definite. Our analysis also uses this property. However, their analysis breaks down due to incorrect assumptions. More specifically, each function f_i is assumed in [18] to be strongly convex and also has a bounded and Lipschitz gradient (i.e., its gradient is bounded and Lipschitz continuous). However, no function can satisfy these assumptions simultaneously since gradients of a strongly convex are strictly increasing and unbounded.

It is worth noting that our algorithm can be applied to a time-varying directed network after a straightforward modification; our convergence proof, however, will need a significant change.

The rest of this note is organized as follows. Section 2 introduces the problem setup and preliminaries. Section 3 develops ExtraPush and Normalized ExtraPush. Section 4 establishes the optimality conditions for ExtraPush and shows its convergence under the boundedness assumption. Section 5 assumes that the objective is strongly convex and shows that Normalized ExtraPush produces a bounded sequence that converges linearly. Section 6 presents our numerical simulation results. We conclude this paper in Section 7.

Notation: Let $\mathbf{I}_n$ denote an identity matrix with the size $n \times n$, and $\mathbf{1}_{n\times p} \in \mathbf{R}^{n\times p}$ denote the *matrix* with all entries equal to 1. We also use $\mathbf{1}_n \in \mathbf{R}^n$ as a vector of all 1's. For any *vector* x, we let x_i denote its ith component and $\mathbf{diag}(x)$ denote the diagonal matrix generated by x. For any matrix X, X^T denotes its transpose, X_{ij} denotes its (i,j)th component, and $\|X\| \triangleq \sqrt{\langle X, X\rangle} = \sqrt{\sum_{i,j} X_{ij}^2}$ denotes its Frobenius norm. The largest and smallest eigenvalues of matrix X are denoted as $\lambda_{\max}(X)$ and $\lambda_{\min}(X)$, respectively. For any matrix $B \in \mathbf{R}^{m\times n}$, $\mathbf{null}(B) \triangleq \{x \in \mathbf{R}^n | Bx = 0\}$ is the null space of B. Given a matrix $B \in \mathbf{R}^{m\times n}$, by $Z \in \mathbf{null}(B)$,