# Communication-Efficient Decentralized Cooperative Data Analytics in Sensor Networks

Liang Zhao[1], Zhihua Li[2(✉)], and Shujie Guo[2]

[1] University of South Carolina Upstate, Spartanburg, SC 29303, USA
lzhao2@uscupstate.edu
[2] Jiangnan University, Wuxi 214122, Jiangsu, China
zhli@jiangnan.edu.cn, 1149165216@qq.com

**Abstract.** This paper presents a novel approach enabling communication-efficient decentralized data analytics in sensor networks. The proposed method aims to solve the decentralized consensus problem in a network such that all the nodes try to estimate the parameters of the global model and they should reach an agreement on the value of the model eventually. Our algorithm leverages broadcasting communication and is performed in a asynchronous manner in the sense that each node can update its estimate independent of others. All the nodes in the network can run the same algorithm in parallel and no synchronization is required. Numerical experiments demonstrate that the proposed algorithm outperforms the benchmark, and it is a promising approach for big data analytics in sensor networks.

**Keywords:** Big data · Data analytics · Decentralized computing
Sensor networks · Asynchronous algorithm

## 1 Introduction

In the era of big data, the goal of transforming big data into actionable insights brings opportunities and also challenges into the community. High volume of data is generated from all over the world everyday. At the same time, data is coming in at a much higher speed, often close to real-time. Thus, there is a huge demand for efficient fast data analyzing approaches. In addition, to analyze big data, big model is always equipped in order to empower deep insights extraction. It is known that many big data analytics problems boil down to: How to apply advanced data analytics programs to large-scale problems with Big Data and Big Model. Essentially, convex optimization is at the core of solving many of these models. Convex optimization has applications in a wide range of disciplines, such as smart grid [1–3], seismic imaging [4,5], and sensor networks [6,7]. Recently, distributed optimization attracts a lot of attention in the optimization and computing society. It has shown potential to be a promising approach for designing scalable big data analytics solution. In general, distributed optimization can be


L. Meng and Y. Zhang (Eds.): MLICOM 2018, LNICST 251, pp. 663–671, 2018.
https://doi.org/10.1007/978-3-030-00557-3_66

categorized into synchronous optimization and asynchronous optimization. In synchronous optimization, each node needs to wait for its slowest neighbor's information in order to proceed. On the contrary, asynchronous optimization can avoid this issues allowing each node to perform its decision independently and locally. Distributed optimization methods for asynchronous models have been designed in [8–10]. In [8,9], the alternating direction method of multipliers (admm) based algorithms have been proposed. Regarding their communication scheme, every node needs to wake up one of its neighbors randomly to exchange information in each iteration. However, the two works are based on unicast, which is much less preferable than broadcast communication, especially in real-world wireless sensor network scenario. Tsitsiklis [10] proposed an asynchronous model for distributed optimization, while in its model each node maintains a partial vector of the global variable. It is different from our goal of decentralized consensus such that each node contains an estimate of the global common interest. The first broadcast-based asynchronous distributed/decentralized consensus method was proposed in [11]. However, the algorithm is designed only for consensus average problem without "real objective function". Nedic [12] first filled this gap by considering general decentralized convex optimization under the asynchronous broadcast setting. It adopted the asynchronous broadcast model in [11] and developed a (sub)gradient-based update rule for its computation. By replacing (sub)gradient computation with full local optimization, an improved algorithm has been designed in terms of the number of communication rounds [13]. In this presenting work, we propose a novel method combing neighbors' (sub)gradient information in order to further speed up the algorithms in [13].

## 2 Problem Formulation

The formulation of the problem investigated in this paper can be described as follows. Consider an undirected connected network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ denotes the node set and $\mathcal{E}$ is the edge set. The size of network is $m = |\mathcal{V}|$ (cardinality of the set $\mathcal{V}$) and two nodes $i, j$ are called neighbors if $(i, j) \in \mathcal{E}$. Assume an objective $F_i : \mathbb{R}^n \to \mathbb{R}$ is only available to each node (sensor or agent) $i$. It is the data and acquisition process at node $i$. The goal is to find the global consensus solution $x \in X$ to minimize the optimization problem as follows.

$$\min_{x \in X} \left\{ F(x) := \sum_{i=1}^{m} F_i(x) \right\}. \tag{1}$$

Solving (1) in (wireless) sensor networks is nontrivial. First, data is generated in a distributed manner and it would be very costly and even infeasible to transmit all the data into a central place for post-processing due to bandwidth and energy constraints. Each node is able to access its local data only. Second, each node needs to exchange information with other nodes in order to obtain the optimal solution for the whole model. However, each node is assumed to communicate with its immediate neighbors only since multi-hop communication

in the presenting application is very expensive and undesriable. In this work, we adopt broadcasting in our communication scheme. Third, the key challenge solving (1) in the network is the potential high communication cost because each node has only partial knowledge of the whole network. Hence, designing a communication-efficient algorithm would make decentralized data analytics in sensor networks feasible.

**Notation.** Let $x \in \mathbb{R}^n$ be a column vector in problem (1), and $x^i \in \mathbb{R}^n$ be the local copy held privately by node $i$ for every $i \in \mathcal{V}$. Without further remark, vectors are all column vectors. Subscript $k$ is outer iteration number, which is also the number of communication.

**Problem Setup.** Each sensor node is assumed to have its local clock that ticks at a user-customized Poisson rate for unit time, which is independent of the clocks of the other nodes. Each node broadcasts its current estimate to its neighbors at each tick of its local clock. During broadcasting, each sensor receives neighbors' information subject to link failures. For example, when node $i$ broadcasts, its neighbor $j$ will receive $i$'s iterate with probability $p_{ij}$. It is equivalent to consider a virtual global clock existing in the network for the algorithm analysis. Since the Poisson clock of each node (suppose rate $= 1$) is independent of each other, it is same as a global clock with Poisson rate $m$. We can then analyze the problem given that in each global iteration only one node broadcasts its value. There are several additional assumptions adopted in this paper as follows.

**Assumption 1.** *The gradient of function $F_i$ is bounded such that $\|\nabla F_i\| \leq G$, where $G > 0$ is some positive number.*

**Assumption 2.** *The solution set of (1) is nonempty. The private local objective function $F_i$, $i \in \mathcal{V}$ is (sub)differentiable and convex.*

**Assumption 3.** *The constraint set $X$ is bounded.*

**Assumption 4.** $\sum_{k=1}^{\infty} \frac{\rho_{i,k}}{k\alpha_{i,k}} < \infty$, $\sum_{k=1}^{\infty} \frac{\beta_{i,k}}{k\alpha_{i,k}} < \infty$ *almost surely.*

## 3 Proposed Algorithm

### 3.1 Local Full Minimization + Neighbor's (sub)gradient

The main computation steps in this proposed algorithm are:

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1-\theta)\, x_{k-1}^i, \\ x_k^i &= P_X\left[\operatorname*{argmin}_x \left\{ \frac{1}{2\alpha_{i,k}}\|x - y_k^i\|^2 + F_i(x) + \rho_{i,k}\left(\sum_{u\in\mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u)^T (x - y_k^i)\right)\right\}\right]. \end{aligned} \tag{2}$$

The algorithm can be summarized as follows.

**Algorithm 1.** Decentralized Cooperative Data Analytics (DCDA) Algorithm

**Input**: Starting point $x_0^1, x_0^2, \cdots, x_0^m$.
**while** *each node* $i$, $i \in \{1, 2, \cdots, m\}$ *asynchronously* **do**
  **if** *node* $i_k$*'s local clock ticks now* **then**
    Node $i_k$ broadcasts its estimate $x_{k-1}^{i_k}$ and (sub)gradient $\tilde{\nabla} F_i(x_{k-1}^{i_k})$ to its neighbors; Node $i$ who receives node $i_k$'s broadcast updates its solution $x_k^i$ based on (2).
  **end**
**end**

*Remark 1.* In (2), neighbor's (sub)gradients $\left(\sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u)^T \left(x - y_k^i\right)\right)$ are incorporated in the update. Node $i$ then computes its next iterate by performing local minimization over all the terms in second equation of (2).

**Theorem 1.** *Let* $\left\{x_k^i\right\}, \forall i \in \mathcal{V}, k \geq 0$ *be the sequence generated by DCDA Algorithm and given that all the assumptions are satisfied. Then we can have:*

$$\sum_{k=1}^{\infty} \frac{1}{k} \|x_{k-1}^i - \bar{x}_{k-1}\| < \infty, \text{ and } \lim_{k \to \infty} \|x_k^i - \bar{x}_k\| = 0 \text{ almost surely.}$$

**Theorem 2.** *Let* $\left\{x_k^i\right\}, \forall i \in \mathcal{V}, k \geq 0$ *be the sequences generated by DCDA Algorithm and given that all the assumptions are satisfied. Then the sequences converges to a same optimal point almost surely for any node* $i$*.*

*Remark 2.* Theorem 1 implicates that all the nodes in the network will reach a consensus on the solution of the global model defined in (1). In addition, Theorem 2 indicates that the consensus solution is optimal. The attack plan for the proof is similar to the counterpart in [13]. The difference between our proposed algorithm and the local optimization based one in [13] is the extra item from neighbor's (sub)gradients $\left(\sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u)^T \left(x - y_k^i\right)\right)$. The main task is to bound this extra item and then we can use the proof framework in [13] to verify Theorems 1 and 2. We leave the details of the proof to the longer report due to page limit of this conference.

## 4 Interpretation of the Proposed Algorithms

### 4.1 Algorithm Interpretation

In this section, we will interpret and show the rationale of proposing DCDA algorithm. Now assume every node $i$ in the network can access all the local objective functions $F_i, i \in \{1, 2, \cdots, m\}$. The optimal strategy for every node $i$ to obtain the solution then becomes as follows.

$$x^i = \underset{x}{\operatorname{argmin}} \sum_{j=1}^{m} F_j(x), \forall i \in \{1, 2, \cdots, m\}. \tag{3}$$

That is, each node can directly try to minimize the summation of all the local objective functions as a "centralized" machine does (assuming all the data is available in this centralized node). To solve (3), we can evaluate a proximal operator as follows [14].

$$x^i = \mathbf{prox}_{\alpha F}(v) = \underset{x}{\operatorname{argmin}} \left\{ \frac{1}{2\alpha} \|x - v\|^2 + F(x) \right\}, \forall i \in \{1, 2, \cdots, m\}, \quad (4)$$

with certain constant $v$ (independent of decision variable $x$) and parameter $\alpha > 0$. Note that each node $i$ can obtain the optimal solution by evaluating (4), and more importantly there is no communication needed between nodes since $F(x)$ contains all the information in the network. However, this is under an ideal scenario (every node $i$ has the knowledge of all the local functions $F_j$) which will not be valid in our setting of decentralized sensor networks. Considering that $F_i$ is only available to node $i$ locally (according to our assumption in this paper), if we replace the term $F(x)$ in (4) with $F_i$ and let $v = y_k^i$ and $\alpha = \alpha_k^i$, the update rule for node $i$ in [13] is then derived as follows.

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1-\theta) x_{k-1}^i, \\ x_k^i &= \mathbf{prox}_{\alpha_{i,k} F_i}\left(y_k^i\right) = \underset{x}{\operatorname{argmin}} \left\{ \frac{1}{2\alpha_{i,k}} \|x - y_k^i\|^2 + F_i(x) \right\}. \end{aligned} \quad (5)$$

Further linearizing $F_i(x)$ in (5) yields Nedic's algorithm as follows [12].

$$\begin{aligned} y_k^i &= \theta x_{k-1}^{i_k} + (1-\theta) x_{k-1}^i, \\ x_k^i &= \underset{x}{\operatorname{argmin}} \left\{ \frac{1}{2\alpha_{i,k}} \|x - y_k^i\|^2 + \left\langle \tilde{\nabla} F_i\left(y_k^i\right), x \right\rangle \right\} \\ &= y_k^i - \alpha_{i,k} \tilde{\nabla} F_i\left(y_k^i\right). \end{aligned} \quad (6)$$

The deduction of the last step in (6) is based on the optimality condition described as follows.

$$\frac{1}{\alpha_{i,k}}\left(x_k^i - y_k^i\right) + \tilde{\nabla} F_i\left(y_k^i\right) = 0.$$

The first step in (5) and (6) takes the weighted average of node $i$'s solution and neighbor $i_k$'s solution which is the most recent broadcast received. This averaging step aims to mix the neighbor's information and enforce consensus of solutions among all the nodes in the network. Next, the proximal step in (5) forces the new solution $x$ to be close to $y_k^i$ (the weighted average) and optimizes the local objective function $F_i$ simultaneously. Parameter $\alpha_{i,k}$ controls the trade-off between the aforementioned two objectives. To speed up the process of decentralized consensus optimization, we are motivated to propose algorithms by adding the following item into the proximal steps in (5).

$$\sum_{u \in \mathcal{N}_i} \tilde{\nabla} F_u(x_{\tau_{u,k}}^u)^T \left(x - y_k^i\right) \quad (7)$$

The term in (7) contains node $i$'s neighbors' (sub)gradient information and it can be seen that (7) is a linear approximation to $\sum_{u\in\mathcal{N}_i} F_u$. Comparing (2) with (5) we can see that in (2) node $i$ is (approximately) optimizing $F_i(x) + \sum_{u\in\mathcal{N}_i} F_u(x)$ while (5) is optimizing local objective function $F_i(x)$ only. Hence, (2) is a better approximation to the ideal case in (4). In order to execute the computations in (2), node $i_k$ needs to broadcast its estimate $x_{k-1}^{i_k}$ and (sub)gradient $\tilde{\nabla} F_i(x_{k-1}^{i_k})$ to its neighbors.

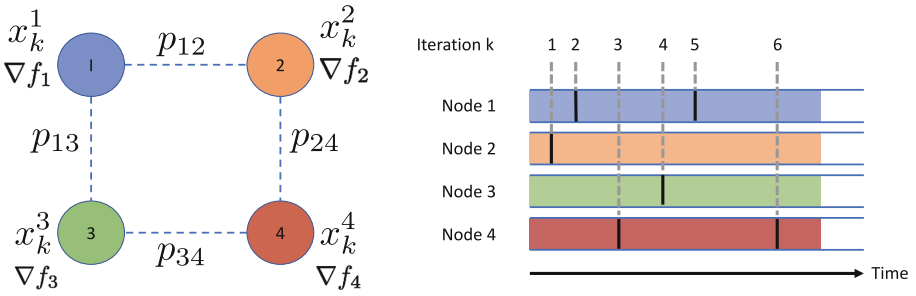


**Fig. 1.** Network model and asynchronous computing. Left: An example of decentralized sensor network. Right: Asynchronous computing model.

### 4.2 An Example of Executing the Proposed Algorithm

We assume the algorithms are performed in a decentralized sensor network illustrated in Fig. 1. There are four nodes in this cyclic network and it is clear to see that $\mathcal{N}_1 = \{2,3\}$, $\mathcal{N}_2 = \{1,4\}$, $\mathcal{N}_3 = \{1,4\}$, $\mathcal{N}_4 = \{2,3\}$. The algorithms run as follows.

**Iteration 1:** Node 2's clock ticks and it broadcasts $x_0^2$ and $\tilde{\nabla} f_2\left(x_0^2\right)$. Node 1 and 4 receive the broadcast and use $x_0^2$ and $\tilde{\nabla} f_2\left(x_0^2\right)$ to update $x_1^1$ and $x_1^4$ based on (2), respectively. Set $x_1^2 \leftarrow x_0^2$, $x_1^3 \leftarrow x_0^3$.

**Iteration 2:** Node 1's clock ticks and it broadcasts $x_1^1$ and $\tilde{\nabla} f_1\left(x_1^1\right)$. Node 2 and 3 receive the broadcast and use $x_1^1$ and $\tilde{\nabla} f_1\left(x_1^1\right)$ to update $x_2^2$ and $x_2^3$ based on (2), respectively. Set $x_2^1 \leftarrow x_1^1$, $x_2^4 \leftarrow x_1^4$.

**Iteration 3:** Node 4's clock ticks and it broadcasts $x_2^4$ and $\tilde{\nabla} f_4\left(x_2^4\right)$. Node 2 and 3 receive the broadcast and use $x_2^4$ and $\tilde{\nabla} f_1\left(x_1^1\right) + \tilde{\nabla} f_4\left(x_2^4\right)$ to update $x_3^2$ and $x_3^3$ based on (2), respectively. Set $x_3^1 \leftarrow x_2^1$, $x_2^4 \leftarrow x_2^4$.

**Iteration 4:** Node 3's clock ticks and it broadcasts $x_3^3$ and $\tilde{\nabla} f_3\left(x_3^3\right)$. Node 1 and 4 receive the broadcast and use $x_3^3$ and $\tilde{\nabla} f_2\left(x_0^2\right) + \tilde{\nabla} f_3\left(x_3^3\right)$ to update $x_4^1$ and $x_4^4$ based on (2), respectively. Set $x_4^2 \leftarrow x_3^2$, $x_4^3 \leftarrow x_3^3$.

It can be seen that after four iterations, each node has gathered all its neighbors' (sub)gradient information. As the algorithm goes on, the neighbors' (sub)gradient information will be updated for each node.

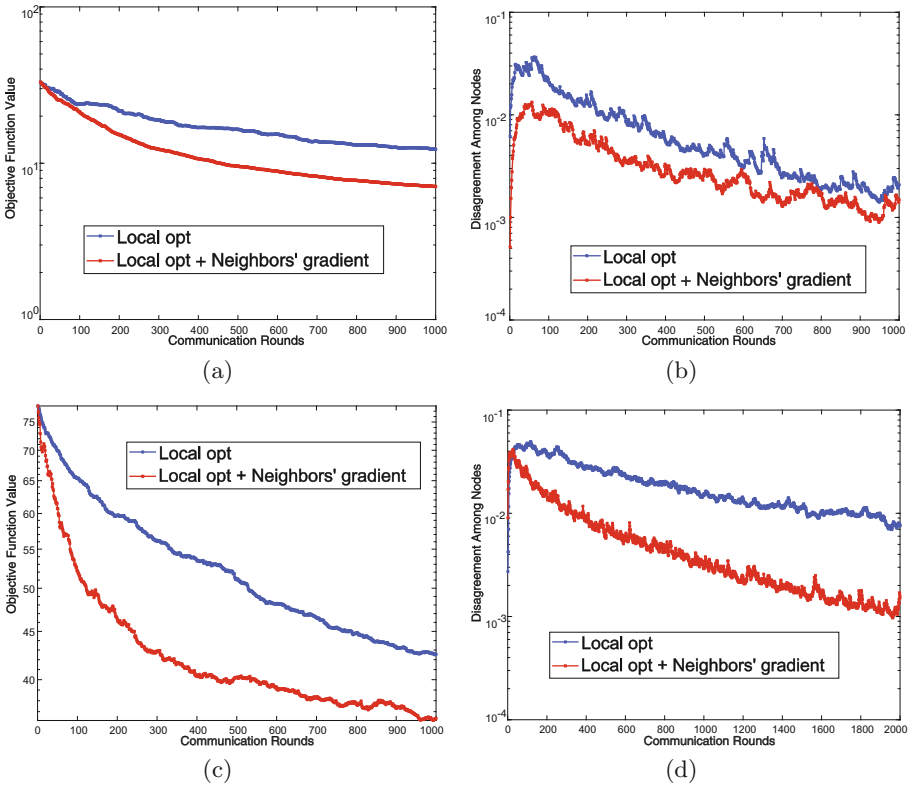


**Fig. 2.** Comparison of convergence speed. Application in decentralized regularized least-squares (a–b). Decentralized logistic regression problem (c–d).

## 5 Numerical Tests

In this section, we test and analyze the performance of the proposed DCDA algorithm. Two types of objective functions are adopted: regularized least-squares and logistic regression. Sensor networks are generated randomly with certain average node degrees (with 200 nodes in total). We investigate the performance of the proposed DCDA algorithm by showing the curve of average objective value and node consensus versus the number of communication rounds.

In decentralized regularized least-squares, node $i$'s local objective function is $F_i(x) = \frac{1}{2}\|\mathbf{A}_i x - \mathbf{b}_i\|_2^2 + \lambda_i \|x\|_2^2$, where the regularization parameter $\lambda_i$ is set to $1/200$, $\mathbf{A}_i$ and $\mathbf{b}_i$ (same dimension for each $i$) are data points available in node $i$. In this scenario, the size of $\mathbf{A}_i$ is $800 \times 3000$ and the dimension of $\mathbf{b}_i$ is set accordingly. In decentralized logistic regression, the local objective function (of node $i$) $F_i$ is set to $F_i(x) = \sum_{j=1}^{p_i} \left( \log \left[ 1 + \exp \left( \left( a_i^j \right)^T x \right) \right] - b_i^j \left( a_i^j \right)^T x \right)$ where $p_i = 10, n = 200$, $\left( a_i^j \right)^T$ represents $j$-th row of $\mathbf{A}_i$ and $b_i^j$ is the $j$-th entry

of $\mathbf{b}_i$. We generate $\mathbf{A}_i \in \mathbb{R}^{p_i \times n}, \forall i$ randomly except the first columns are set to 1. Binary vector $\mathbf{b}_i \in \mathbb{R}^{p_i}$ is generated randomly.

In Fig. 2, we compare the performance of the proposed DCDA with the decentralized algorithm in [13]. It is clear that DCDA outperforms the benchmark in both applications, in terms of the speed to reach optimal objective function value as well as consensus among the nodes in the network.

## 6 Conclusion

We proposed a broadcast-based asynchronous decentralized optimization mechanism for data analytics in sensor networks. Our mechanism leverages the computational capability of each node and let all the nodes in the network cooperate to solve the problem of big data analytics with big model. Our future work includes evaluation of the proposed algorithm using more realistic measures.

## References

1. Zhao, L., Song, W.Z., Tong, L., Wu, Y.: Monitoring for power-line change and outage detection in smart grid via the alternating direction method of multipliers. In: 2014 28th International Conference on Advanced Information Networking and Applications Workshops, pp. 342–346, May 2014
2. Zhao, L., Song, W.Z.: A new multi-objective microgrid restoration via semidefinite programming. In: 2014 IEEE 33rd International Performance Computing and Communications Conference (IPCCC), pp. 1–8, December 2014
3. Zhao, L., Song, W.Z., Tong, L., Wu, Y., Yang, J.: Topology identification in smart grid with limited measurements via convex optimization. In: 2014 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA), pp. 803–808, May 2014
4. Zhao, L., Song, W.Z., Ye, X.: Fast decentralized gradient descent method and applications to in-situ seismic tomography. In: 2015 IEEE International Conference on Big Data (Big Data), pp. 908–917, October 2015
5. Zhao, L., Song, W.-Z., Shi, L., Ye, X.: Decentralised seismic tomography computing in cyber-physical sensor systems. Cyber-Phys. Syst. **1**(2–4), 91–112 (2015)
6. Zhao, L., Song, W.-Z.: Distributed power-line outage detection based on wide area measurement system. Sensors **14**(7), 13114–13133 (2014)
7. Zhao, L., Song, W.Z.: Decentralized consensus in distributed networks. Int. J. Parallel Emergent Distrib. Syst. 1–20 (2016)
8. Wei, E., Ozdaglar, A.: On the o(1/k) convergence of asynchronous distributed alternating direction method of multipliers. arXiv:1307.8254 (2013)
9. Ciblat, P., Iutzeler, F., Bianchi, P., Hachem, W.: Asynchronous distributed optimization using a randomized alternating direction method of multipliers. arXiv:1303.2837 (2013)
10. Tsitsiklis, J.N., Bertsekas, D.P., Athans, M.: Distributed asynchronous deterministic and stochastic gradient optimization algorithms. IEEE Trans. Autom. Control **31**(9), 803–812 (1986)
11. Aysal, T.C., Yildiz, M.E., Sarwate, A.D., Scaglione, A.: Broadcast gossip algorithms for consensus. IEEE Trans. Signal Process. **57**(7), 2748–2761 (2009)

12. Nedic, A.: Asynchronous broadcast-based convex optimization over a network. IEEE Trans. Autom. Control **56**(6), 1337–1351 (2011)
13. Zhao, L., Song, W.-Z., Ye, X., Gu, Y.: Asynchronous broadcast-based decentralized learning in sensor networks. Int. J. Parallel Emergent Distrib. Syst. 1–19 (2018)
14. Parikh, N., Boyd, S.: Proximal algorithms. Found. Trends Optim. **1**(3), 127–239 (2014)