CPCA: A Chebyshev Proxy and Consensus based Algorithm for General Distributed Optimization

Zhiyu He, Jianping He, Cailian Chen and Xinping Guan

Abstract— **We consider a general distributed optimization problem, aiming to optimize the average of a set of local objectives that are Lipschitz continuous univariate functions, with the existence of same local constraint sets. To solve the problem, we propose a Chebyshev Proxy and Consensus-based Algorithm (CPCA). Compared with existing distributed optimization algorithms, CPCA is able to address the problem with non-convex Lipschitz objectives, and has low computational costs since it is free from gradient or projection calculations. These benefits result from i) the idea of optimizing a Chebyshev polynomial approximation (i.e. a proxy) for the global objective to obtain (ϵ)-optimal solutions for any given precision (ϵ), and ii) the use of average consensus where the local proxies' coefficient vectors are gossiped to enable every agent to obtain such a global proxy. We provide comprehensive analysis of the accuracy and complexities of the proposed algorithm. Simulations are conducted to illustrate its effectiveness.**

I. Introduction

Distributed optimization algorithms enable multiple agents in a network to collaboratively solve the problem of optimizing the average of local objective functions. Their developments have been motivated by wide application scenarios, including distributed machine learning [1], statistics [2], estimation [3] and coordination [4]–[6] in large-scale multi-agent systems, like wireless sensor networks, smart grids, multi-robot systems and so on.

Most distributed optimization algorithms are first-order methods based on gradients, and can be divided into three categories: primal, dual and primal-dual methods. They all employ consensus to make all agents' local estimates be close after iterations. Primal methods [7]–[10] use sub-gradient descent to drive the estimates to the optimal points in the primal domain. Dual methods [11], [12] consider the dual of the problem with consensus equality constraints and go on iterations in the dual domain. Primal-dual methods [13]–[15] update primal and dual variables in parallel, so as to reach the saddle points of the Lagrangian. When there also exist local set constraints, projection operations onto the constraint sets can be taken to extend existing unconstrained methods [16], [17].

However, existing iterative gradient-based algorithms generally work under the premise that the objective functions are convex, and involve computationally expensive operations at every iteration. They need convexity assumptions either to guarantee the reach of the global rather than local optimizer, or to make sure the hold of strong duality. Since they require every agent at every iteration to compute gradients or optimize a local sub-problem, which can be costly in general, the total computational costs are rather high when the iterations go long. Moreover, constrained methods have extra costs of calculating projections and still suffer from sub-linear convergence rates. It still remains an open problem how to design algorithms that can handle problems with non-convex objectives and are computationally inexpensive.

Recently, there have been works in the numerical analysis field that use the Chebyshev polynomial approximation to substitute for the target function defined on an interval, so as to make the study of its property much easier [18]–[20]. Since an arbitrarily precise approximation (i.e. a proxy) can be constructed for any continuous objective on the entire interval, the difference between their optimal values can be arbitrarily small. This means that we can turn to solve the easier problem of optimizing the proxy of the global objective instead. In addition, the coefficient vector of the proxy serves as a discrete representation of it. This implies that by going on average consensus, agents can obtain the average of all those vectors of the local proxies. This average is exactly the representation of the proxy of the global objective. Agents can then compute the optimum of the polynomial recovered from this vector by evaluating it on all its critical points, and thus obtain close estimates for that of the global objective.

Based on these intuitions, we develop a Chebyshev Proxy and Consensus-based Algorithm (CPCA) for the constrained distributed optimization problem. This problem has same local constraint sets (the extension to the case with different sets is not difficult) and general Lipschitz continuous univariate objectives. The main contributions are summarized as follows:

1) To the best of our knowledge, this is the first work to achieve global optimum for constrained distributed optimization problems without convexity assumptions on the objectives.
2) We propose a novel algorithm, CPCA, based on Chebyshev polynomial approximation and consensus. It jumps out of the scope of iterative gradient- and projection-based methods, and solves an easier problem of optimizing the polynomial approximation for the global objective instead. Also, it has a simple iterating structure of processing coefficient vectors, rather than gossiping estimates of the optimizers followed by additional operations.

The authors are with the Department of Automation, Shanghai Jiao Tong University, and Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China. Emails: {hzy970920, jphe, cailianchen, xpguan}@sjtu.edu.cn. This research work is partially sponsored by the National Key R&D Program of China 2017YFE0114600, and NSFC of China (61973218, 61828301, 61622307).

3) We provide comprehensive analysis of the accuracy and the complexities of CPCA. Specifically, for any given error tolerance ϵ, the solutions it yields are ϵ-optimal. The communication and zero-order oracle complexities for every agent are $\mathcal{O}(N\log(N\log m/\epsilon))$ and $\mathcal{O}(m)$ respectively, where N is the number of agents and m is the degree of the approximation. Note that our algorithm is free from gradient computations, and has oracle complexities independent of N and ϵ. This shows that our algorithm are computationally inexpensive.

The rest of this paper is organized as follows. Section II provides some preliminaries and formally defines the problem of interest. Section III presents our algorithm, namely CPCA. Section IV provides the analysis of the accuracy and complexities of CPCA. Section V shows the simulation results. Finally, Section VI concludes this paper.

Notations. For vector a, we use a' to denote its transpose, $\|a\|$ to denote its l_2-norm, and $\|a\|_\infty$ to denote its l_∞-norm. We denote the all-ones vector by $\mathbf{1}$. The superscript t denotes the index of the iterations, and the subscripts i, j denote the indexes of the agents. The scripts in parentheses k, l denote the indexes of the elements in a vector.

II. Preliminaries and Problem Formulation

Consider a network with N agents. Its communication topology is described as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of agents, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. It is noted that agent j can receive information from agent i if and only if (*iff*) $(i, j) \in \mathcal{E}$. Throughout the paper, we assume that $\mathcal{G}$ is a static, connected and undirected graph.

A. Typical Types of Consensus

Suppose every agent i maintains a local variable $x_i^t \in \mathbb{R}$, where $t \in \mathbb{N}$ is the number of iterations. The maximum consensus protocol is

$$x_i^t = \max\left\{x_i^{t-1},\ \max_{j\in\mathcal{N}_i} x_j^{t-1}\right\}, \tag{1}$$

where $\mathcal{N}_i$ denotes the set of agent i's neighbors. By (1), all x_i^t converge to $\max_{i\in\mathcal{V}} x_i^0$ in T ($T \le D$) iterations, where D is the diameter of $\mathcal{G}$.

Another type of consensus is average consensus. There have already been such protocols, e.g., [21], whose convergence time scale quadratically with N. Recently, [22] shows that the order of convergence in terms of N can be brought down to linear with the following assumption.

Assumption 1 ([22]). *Every agent i in $\mathcal{G}$ knows an upper bound U on N, such that $\exists\, c \in \mathbb{R} : N \le U \le cN$.*

The linear time average consensus protocol [22] is

$$\begin{cases} y_i^t = x_i^{t-1} + \dfrac{1}{2}\displaystyle\sum_{j\in\mathcal{N}_i} \dfrac{x_j^{t-1} - x_i^{t-1}}{\max(d_i, d_j)}, \\ x_i^t = y_i^t + \left(1 - \dfrac{2}{9U+1}\right)(y_i^t - y_i^{t-1}), \end{cases} \tag{2}$$

where y_i^0 is initialized to be x_i^0, and d_i is the degree of node i in $\mathcal{G}$. Let $x^t = [x_1^t, \ldots, x_N^t]'$, $y^t = [y_1^t, \ldots, y_N^t]'$. The convergence of (2) is discussed in the following theorem.

Theorem 1 ([22]). *If Assumption 1 holds, with (2), we have*

$$\left\|y^t - \bar{x}\mathbf{1}\right\| \le \sqrt{2}\rho^t \left\|y^0 - \bar{x}\mathbf{1}\right\|,$$

where $\bar{x} = 1/N\sum_{i=1}^N x_i^0$, and $\rho = \sqrt{1 - 1/(9U)}$.

Let $e^t = y^t - \bar{x}\mathbf{1}$. We derive the decaying property of $\max_{k,l}|e^t(k) - e^t(l)|$, where k and l are the indexes of the elements in e^t.

Corollary 1. *If Assumption 1 holds, with (2), we have*

$$\max_{k,l}\left|e^t(k) - e^t(l)\right| \le 2\sqrt{2N}\rho^t \max_{k,l}\left|e^0(k) - e^0(l)\right|.$$

Proof. By rewriting (2) as matrix equations and noting that the weight matrix is doubly stochastic, we have $\mathbf{1}'y^t = \mathbf{1}'x^0$, $\forall t \in \mathbb{N}_+$. [1] Hence, $\mathbf{1}'e^t = 0$, $\forall t \in \mathbb{N}_+$. As a result, $\|e^t\|_\infty \le \max_{k,l}|e^t(k) - e^t(l)| \le 2\|e^t\|_\infty$. Therefore,

$$\begin{aligned} \max_{k,l}\left|e^t(k) - e^t(l)\right| &\le 2\left\|e^t\right\|_\infty \le 2\left\|e^t\right\|_2 \le 2\sqrt{2}\rho^t\left\|e^0\right\|_2 \\ &\le 2\sqrt{2N}\rho^t\left\|e^0\right\|_\infty \le 2\sqrt{2N}\rho^t \max_{k,l}\left|e^0(k) - e^0(l)\right|. \end{aligned}$$

□

B. Chebyshev Polynomial Approximation

For a Lipschitz continuous function $g(x)$ with $x \in [a, b]$, its degree m Chebyshev interpolant $p_m(x)$ is

$$p_m(x) = \sum_{j=0}^{m} c_j T_j\left(\frac{2x - (a+b)}{b-a}\right), \quad x \in [a, b], \tag{3}$$

where $T_j(\cdot)$ denotes the j-th Chebyshev polynomial. As m grows, $p_m(x)$ converges to $g(x)$ uniformly on the given interval [18], i.e.,

$$\forall x \in [a, b],\ \ |p_m(x) - g(x)| \to 0,\ \text{as } m \to \infty.$$

Let c_j' be the Chebyshev coefficients of the derivative of $p_m(x)$. Then, we have the following recurrence formula,

$$c_j' = \begin{cases} 0, & j = m, m+1, \ldots \\ c_{j+2}' + 2(j+1)Sc_{j+1}, & j = m-1, \ldots, 1 \\ c_2'/2 + Sc_1, & j = 0, \end{cases} \tag{4}$$

where $S = 2/(b-a)$.

By [18], the roots of $p_m(x)$ are the eigenvalues of a colleague matrix M_C, which is given by

$$\begin{pmatrix} 0 & 1 & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{2} & 0 & \frac{1}{2} \\ -\frac{c_0}{2c_m} & -\frac{c_1}{2c_m} & \cdots & \frac{1}{2} - \frac{c_{m-2}}{2c_m} & -\frac{c_{m-1}}{2c_m} \end{pmatrix}_{m\times m} \tag{5}$$

Clearly, M_C is a sparse matrix whose non-zero elements are trivial functions of the Chebyshev coefficients of $p_m(x)$.

[1] A detailed proof can be found in the proof of Theorem 2.

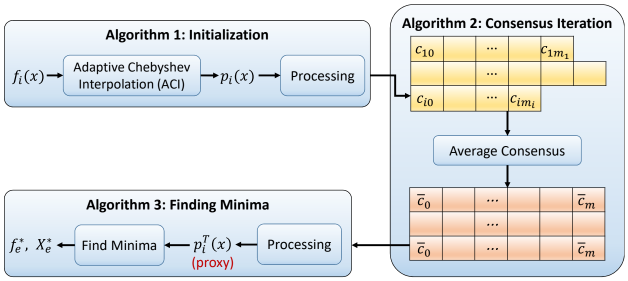


Fig. 1. The Architecture of CPCA

C. Problem Formulation

In this paper, we mainly focus on the following constrained distributed optimization problem

$$\begin{aligned} \min_x \quad & f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x), \\ \text{s.t.} \quad & x \in X = \bigcap_{i=1}^{N} X_i, \end{aligned} \tag{6}$$

where $f_i(x) : X_i \to \mathbb{R}$ is the local objective function and is known only by agent i, and $X_i \subseteq \mathbb{R}$ is the local constraint set. Some basic assumptions are given as follows.

Assumption 2. *Every $f_i(x)$ is Lipschitz continuous on X_i.*

The Lipschitz continuous assumption on local objective functions is quite common among the literature [12], [13]. What distinguishes our work from others is that we require neither global nor local objectives to be convex.

Assumption 3. *All X_i are the same closed, bounded and convex set.*

By Assumption 3, for all $i \in \mathcal{V}$, X_i is the closed interval $[a, b]$, where $a, b \in \mathbb{R}$. We make this simplification to highlight the design and the structure of our algorithm.

III. Algorithm Development

In this section, we present CPCA to solve problem (6) distributively. Figure 1 provides an overview of our algorithm.

A. Initialization: Construction of Approximations

In the initialization stage, every agent first constructs the approximating polynomial, $p_i(x)$, corresponding to $f_i(x)$ on the given interval, such that

$$|f_i(x) - p_i(x)| \le \epsilon_1, \quad \forall x \in [a, b]. \tag{7}$$

Then, it obtains a local vector p_i^0 storing the information of the Chebyshev coefficients with additional computations. The details are summarized in Algorithm 1.

In Algorithm 1, $p_i(x)$ is constructed by using Adaptive Chebyshev Interpolation (ACI) [20]. The key insight is to systematically increase the degree of the Chebyshev interpolant until certain stopping criterion (proposed in [20]) is met. Agents then calculate the Chebyshev coefficients $\{c_j\}$ of the derivative of $p_i(x)$ with (4). The m_i-dimensional local vector p_i^0 stores these coefficients as well as the first

Algorithm 1 Initialization of CPCA

Input: $f_i(x), \epsilon_1$ and $X = [a, b]$.
Output: p_i^0.

1: **Initialize** $m_i = 2$.

2: Calculate $S_{m_i} = \{x_j\}, \{f_j\}$ according to

$$\begin{cases} x_j = \dfrac{b-a}{2}\cos\left(\dfrac{j\pi}{m_i}\right) + \dfrac{a+b}{2}, \\ f_j = f_i(x_j), \end{cases}$$

where $j = 0, 1, \ldots, m_i$.

3: Calculate $\{c_k\}$ according to

$$c_k = \frac{2}{m_i}\sum_{j=0}^{m_i}{}'' f_j \cos\left(\frac{kj\pi}{m_i}\right),$$

where the double prime indicates that the first and the last terms are halved, and $k = 0, 1, \ldots, m_i$.

4: If

$$\max_{x_j \in \left(S_{2m_i} - S_{m_i}\right)} |f_i(x_j) - p_i(x_j)| \le \epsilon_1,$$

where $p_i(x)$ is in the form of (3) with c_k being the coefficients, then go to step 5. Or set $m_i \leftarrow 2m_i$ and go to step 2.

5: Calculate $\{c'_k | k = 0, \ldots, m_i - 1\}$ with (4).

6: Set $p_i^0 = [c_0, c'_0, c'_1, \ldots, c'_{m_i-1}]'$.

coefficient c_0 of $p_i(x)$, and will be exchanged and updated afterward.

ACI is robust and can yield $p_i(x)$ satisfying (7) for most Lipschitz continuous $f_i(x)$ that are analytic or have several orders of derivatives [20]. Therefore, we can conclude that $p_i(x)$ obtained from Algorithm 1 satisfies (7).

B. Iteration: Consensus-based Update of Local Vectors

In the iteration stage, agents update their local vectors based on consensus, so as to make them converge to the average of all the initial values. The goal is to ensure that when the iteration stage ends,

$$\max_{i \in \mathcal{V}} \left\| p_i^K - \bar{p} \right\|_\infty \le \delta$$

holds, where K is the number of iterations, p_i^K is the local vector of agent i and $\bar{p} = 1/N \sum_{i=1}^{N} p_i^0$ is the average. To meet the requirement of precision, δ is set by

$$\delta = \frac{\epsilon_2}{1 + \frac{1}{S}\left(\ln m + \frac{3}{2}\right)}. \tag{8}$$

Note that δ is proportional to the given precision ϵ_2, with $m = \max_{i \in \mathcal{V}} m_i$ and $S = 2/(b-a)$. The details are summarized in Algorithm 2.

Remark 1. *After the initialization, different agents have p_i^0 of different dimensions $m_i + 1$. While going on the update in (9) and (10), agents ensure agreements in dimension by aligning and padding zeros to lower dimensional local vectors involved. After finite number of iterations (less than the diameter of $\mathcal{G}$, certainly less than U), all the local vectors will be of the same dimension $\max_{i \in \mathcal{V}} m_i + 1 = m + 1$.*

In Algorithm 2, local vectors p_i^t are updated based on the linear time average consensus in (2), and will converge to $\bar{p}$. There are also two auxiliary vectors, r_i^t and s_i^t, updated

Algorithm 2 Iteration of CPCA

Input: p_i^0, ϵ_2, U and $X = [a, b]$.
Output: p_i^K.

Initialize: $q_i^0 = r_i^0 = s_i^0 = p_i^0,\ K = U,\ S = 2/(b-a),\ \rho = \sqrt{1 - 1/(9U)}$.
while $t \le K$ **do**
  Update p_i^t and q_i^t according to

$$\begin{cases} p_i^t = q_i^{t-1} + \dfrac{1}{2} \sum_{j \in \mathcal{N}_i} \dfrac{q_j^{t-1} - q_i^{t-1}}{\max(d_i, d_j)}, \\ q_i^t = p_i^t + \left(1 - \dfrac{2}{9U+1}\right)(p_i^t - p_i^{t-1}). \end{cases} \tag{9}$$

  Update the l-th element of r_i^t and s_i^t according to (1).

$$r_i^t(l) = \max_{j \in \mathcal{N}_i \cup \{i\}} r_j^{t-1}(l), \quad s_i^t(l) = \min_{j \in \mathcal{N}_i \cup \{i\}} s_j^{t-1}(l) \tag{10}$$

  if $t = K$ **and** $t = U$ **then**
    Set

$$K \leftarrow \max\left(\left\lceil \frac{\ln(\delta / 2\sqrt{2U} \|r_i^t - s_i^t\|_\infty)}{\ln \rho} \right\rceil, U \right), \tag{11}$$

  end if
  Set $t \leftarrow t + 1$.
end while
return p_i^K.

according to the maxi/minimum consensus in (1) within the same iteration. After U iterations, agents obtain the initial maximum deviation $d_0 := \max_{i,j \in \mathcal{V}} \|p_i^0 - p_j^0\|_\infty = \|r_i^U - s_i^U\|_\infty$, and then calculate an upper bound K on the number of iterations needed to guarantee specific average consensus precision. After going on K iterations, agents terminate simultaneously. The following theorem demonstrates the effectiveness of Algorithm 2.

Theorem 2. *For Algorithm 2, we have*

$$\max_{i \in \mathcal{V}} \left\| p_i^K - \bar{p} \right\|_\infty \le \delta, \tag{12}$$

where δ satisfies (8).

Proof. First, we show that $\sum_{i \in \mathcal{V}} p_i^t = \sum_{i \in \mathcal{V}} p_i^0,\ \forall t \in \mathbb{N}_+$. It follows from Remark 1 that all p_i^t will be of the same dimension in finite time. To simplify the analysis, we assume that the aligning and zero-padding mentioned in Remark 1 are completed in advance and that $p_i^0 \in \mathbb{R}^{m+1},\ \forall i \in \mathcal{V}$. Let $P^t = [p_1^t, \ldots, p_N^t]'$ and $Q^t = [q_1^t, \ldots, q_N]'$. By introducing the lazy Metropolis weight matrix W, we rewrite (9) as

$$\begin{cases} P^t = W Q^{t-1}, \\ Q^t = P^t + \left(1 - \dfrac{2}{9U+1}\right)\left(P^t - P^{t-1}\right), \end{cases}$$

where P^0 and Q^0 are initialized to be equal. Since W is doubly stochastic, $\mathbf{1}'P^1 = \mathbf{1}'WQ^0 = \mathbf{1}'Q^0 = \mathbf{1}'P^0$. Assume that for $t = k,\ \mathbf{1}'P^k = \mathbf{1}'P^{k-1}$. Since

$$\mathbf{1}'Q^k = \mathbf{1}'P^k + \left(1 - \frac{2}{9U+1}\right)\mathbf{1}'\left(P^k - P^{k-1}\right) = \mathbf{1}'P^k,$$

we have $\mathbf{1}'P^{k+1} = \mathbf{1}'WQ^k = \mathbf{1}'Q^k = \mathbf{1}'P^k$. Therefore, $\mathbf{1}'P^t = \mathbf{1}'P^{t-1} = \mathbf{1}'P^0,\ \forall t \in \mathbb{N}_+$. This implies that $\sum_{i=1}^N p_i^t = \sum_{i=1}^N p_i^0,\ \forall t \in \mathbb{N}_+$.

Then, we derive an upper bound for the left-hand side of (12). For any $i \in \mathcal{V}$ and $t \in \mathbb{N}_+$, we have

$$\begin{aligned} \left\|p_i^t - \bar{p}\right\|_\infty &= \left\| p_i^t - \frac{1}{N}\sum_{j=1}^N p_j^t \right\|_\infty = \left\| \frac{1}{N}\sum_{j=1}^N \left(p_i^t - p_j^t\right) \right\|_\infty \\ &\le \frac{1}{N}\sum_{j=1}^N \left\|p_i^t - p_j^t\right\|_\infty \le \frac{1}{N}\sum_{j=1}^N \max_{i,j \in \mathcal{V}} \left\|p_i^t - p_j^t\right\|_\infty \\ &= \max_{i,j \in \mathcal{V}} \left\|p_i^t - p_j^t\right\|_\infty \triangleq d_t. \end{aligned}$$

Subsequently, we characterize the decaying property of d_t. Suppose that at the t-th iteration, for some pair of $i, j \in \mathcal{V},\ |p_i^t(k) - p_j^t(k)| = d_t$. Based on Corollary 1, we have

$$\begin{aligned} d_t &= \left|p_i^t(k) - p_j^t(k)\right| = \left|e_i^t(k) - e_j^t(k)\right| \\ &\le 2\sqrt{2N}\rho^t \left|e_i^0(k) - e_j^0(k)\right| \le 2\sqrt{2N}\rho^t \max_{i,j \in \mathcal{V}} \left\|e_i^0 - e_j^0\right\|_\infty \\ &= 2\sqrt{2N}\rho^t \max_{i,j \in \mathcal{V}} \left\|p_i^0 - p_j^0\right\|_\infty \le 2\sqrt{2U}\rho^t d_0. \end{aligned}$$

Finally, we prove (12). By (11), Algorithm 2 will at least go on $\lceil \ln(\delta / 2\sqrt{2U} d_0) / \ln(\rho) \rceil$ iterations. Therefore,

$$\left\|p_i^T - \bar{p}\right\|_\infty \le d_T \le 2\sqrt{2U}\rho^T d_0 \le \delta. \qquad \square$$

Algorithm 3 Finding Minima of CPCA

Input: p_i^K and $X = [a, b]$.
Output: f_e^* and X_e^*.

Construct M_C based on $\{p_i^K(j) | j = 2, \ldots, m+1\}$ by (5).
Calculate $\{c_j | j = 0, \ldots, m\}$ according to

$$c_j = \begin{cases} p_i^K(1), & j = 0, \\ \dfrac{1}{2S}\left(2p_i^K(2) - p_i^K(4)\right), & j = 1, \\ \dfrac{1}{2jS}\left(p_i^K(j+1) - p_i^K(j+3)\right), & j = 2, \ldots, m, \end{cases}$$

where $S = 2/(b-a)$, and $p_i^K(j)$ is defined as 0 for j out of range.
Calculate the real eigenvalues $E = \{\lambda_j\}$ of M_C that lie in $[a, b]$.
Set

$$f_e^* = \min_{x \in X_K} p_i^K(x), \quad X_e^* = \arg\min_{x \in X_K} p_i^K(x),$$

where $X_K = E \cup \{a, b\}$, and

$$p_i^K(x) = \sum_{j=0}^m c_j T_j\left(\frac{2x - (a+b)}{b-a}\right), \quad x \in [a, b].$$

C. Finding Minima: Taking a Straightforward Approach

In this stage, agents optimize the polynomial proxy $p_i^K(x)$ recovered from p_i^K independently, and thus obtain ϵ-optimal solutions for (6). This is done by first finding the stationary points of $p_i^K(x)$, then taking the minimum of all the critical values of it. The details are summarized in Algorithm 3.

In Algorithm 3, agent i first prepares the colleague matrix M_C of order $(m-1)$ based on all but the first elements of p_i^K. Then, it calculates $\{c_j\}$ based on a reverse recurrence

Algorithm 4 CPCA

Input: ϵ, U and $X = [a, b]$.
Output: $f_e^*,\ X_e^*$ for agent i $(i \in \mathcal{V})$.
1: **for each** agent $i \in \mathcal{V}$ **do**
2: Take $f_i(x), X$ and $\epsilon_1 = \frac{\epsilon}{2}$ as inputs, and run **Algorithm 1**.
3: Take p_i^0 returned in step 2, $\epsilon_2 = \frac{\epsilon}{2}$, U and X as inputs, and run **Algorithm 2**.
4: Take $p_i^(K)$ returned in step 3 and X as inputs, and run **Algorithm 3**.
5: Return f_e^* and X_e^* obtained in step 4.
6: **end for**

formula of (4) to obtain the proxy $p_i^K(x)$. Note that the Chebyshev coefficients of the derivative of $p_i^K(x)$ are stored in p_i^K. It can therefore computes the real eigenvalues of M_C to obtain the real roots of the derivative, i.e., the stationary points of $p_i^K(x)$. Finally, it evaluates $p_i^K(x)$ on X_K, the set of all the critical points, and gets the minimum value f_e^* of $p_i^K(x)$.

D. CPCA

CPCA is composed of the three algorithms discussed previously. It is formulated as Algorithm 4.

Note that our algorithm takes the given error tolerance ϵ as one of the inputs. This tolerance ϵ is used to set some key parameters $\epsilon_1,\ \epsilon_2$ (and also δ) utilized in the corresponding algorithms. As a result, the reach of ϵ-optimality of CPCA can be guaranteed.

IV. Analysis of CPCA

A. Accuracy of CPCA

We first provide a lemma stating that if two functions f and g are sufficiently close on the entire interval, their minimum values $f(x_f^*)$ and $g(x_g^*)$ are sufficiently close also.

Lemma 3. *Suppose f, g satisfy $|f(x) - g(x)| \le \epsilon,\ \forall x \in [a, b]$. Then, $\left|f(x_f^*) - g(x_g^*)\right| \le \epsilon$.*

Proof. First, we have $-\epsilon \le f(x_f^*) - g(x_f^*) \le \epsilon$, and $-\epsilon \le f(x_g^*) - g(x_g^*) \le \epsilon$. It follows that $f(x_f^*) \le f(x_g^*) \le g(x_g^*) + \epsilon$, and $g(x_g^*) \le g(x_f^*) \le f(x_f^*) + \epsilon$. This leads to $-\epsilon \le f(x_f^*) - g(x_g^*) \le \epsilon$. □

The accuracy of CPCA is established as follows. We use ϵ, f^* to denote the given error tolerance, the optimal value of problem (6), respectively.

Theorem 4. *Suppose that Assumptions 1-3 hold. If ϵ is specified, CPCA ensures that every agent obtains ϵ-optimal solutions f_e^* for problem (6), i.e., $|f_e^* - f^*| \le \epsilon$.*

Proof. We first establish the closeness between $p_i^K(x)$ and $\bar{p}(x)$. Let $e_i^K = p_i^K - \bar{p}$. By Theorem 2, $\|e_i^K\|_\infty \le \delta$. Suppose the Chebyshev coefficients of $p_i^K(x)$ and $\bar{p}(x)$ are $\{c_j\}$ and $\{\bar{c}_j\}$, respectively. Then,

$$|c_j - \bar{c}_j| \le \begin{cases} |e_i^K(1)| \le \delta, & j = 0, \\ \dfrac{|2e_i^K(2) - e_i^K(4)|}{2S} \le \dfrac{3\delta}{2S}, & j = 1, \\ \dfrac{|e_i^K(j+1) - e_i^K(j+3)|}{2jS} \le \dfrac{\delta}{jS}, & j = 2, \ldots, m. \end{cases}$$

Consequently,

$$|p_i^K(x) - \bar{p}(x)| \le \sum_{j=0}^{m} |c_j - \bar{c}_j| \cdot 1 < \delta\Big(1 + \frac{1}{S}\Big(\frac{1}{2} + \sum_{j=1}^{m} \frac{1}{m}\Big)\Big)$$
$$< \delta\Big(1 + \frac{1}{S}\big(\frac{3}{2} + \ln m\big)\Big) \le \epsilon_2,$$

where we use the fact that $|T_j(x)| \le 1, \forall x \in [-1, 1]$.

Then, we establish the closeness between $\bar{p}(x)$ and $f(x)$. Since $\bar{p}$ is the average of all p_i^0, $\bar{p}(x)$ is also the average of all $p_i(x)$. Based on the results in Section III-A, we have

$$|\bar{p}(x) - f(x)| = \left|\frac{1}{N}\sum_{i=1}^{N}\big(p_i(x) - f_i(x)\big)\right|$$
$$\le \frac{1}{N}\sum_{i=1}^{N}|p_i(x) - f_i(x)| \le \frac{1}{N}N\epsilon_1 = \epsilon_1.$$

Note that $\epsilon_1 = \epsilon_2 = \epsilon/2$. Hence,

$$\left|p_i^K(x) - f(x)\right| \le \left|p_i^K(x) - \bar{p}(x)\right| + |\bar{p}(x) - f(x)|$$
$$\le \epsilon_1 + \epsilon_2 = \epsilon.$$

Based on Lemma 3, we have $|f_e^* - f^*| \le \epsilon$. □

TABLE I
Complexities of CPCA

Stages	Operations	Oracles	Communications
init	$\mathcal{O}(m^2 \log m)$	$\mathcal{O}(m)$	0
iteration	$\mathcal{O}\Big(N \log\big(\frac{N \log m}{\epsilon}\big)\Big)$	0	$\mathcal{O}\Big(N \log\big(\frac{N \log m}{\epsilon}\big)\Big)$
solve	$\mathcal{O}(m^3)$	0	0
whole	$\mathcal{O}\Big(N \log\big(\frac{N \log m}{\epsilon}\big)\Big)$	$\mathcal{O}(m)$	$\mathcal{O}\Big(N \log\big(\frac{N \log m}{\epsilon}\big)\Big)$

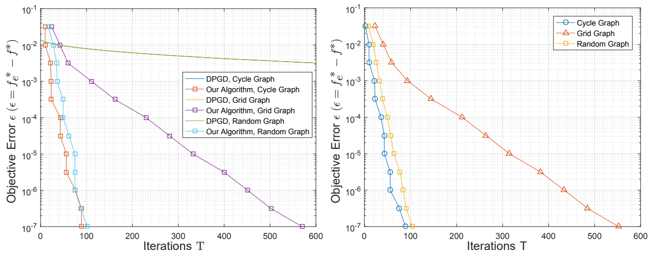
B. Complexity of CPCA

We provide the analysis of the computational and communication complexities of CPCA. The following theorem establishes the total number of elementary operations (additions and multiplications), zero-order oracle queries (calls for objective functions' values) and inter-agent communication needed while running CPCA from a single agent's perspective. The details are summarized in Table I.

Theorem 5. *With CPCA, for every agent, the computational costs are $\mathcal{O}\Big(N \log\big(\dfrac{N \log m}{\epsilon}\big)\Big)$ elementary operations and $\mathcal{O}(m)$ zero-order oracle queries, and the number of communication rounds is $\mathcal{O}\Big(N \log\big(\dfrac{N \log m}{\epsilon}\big)\Big)$.*

Proof. The proof is omitted due to the space limit. □

Note that m is decided by the properties of the objectives, and is independent of N and ϵ. The first-order and zero-order oracle complexities of CPCA are 0 and $\mathcal{O}(m)$, respectively, and is independent of the iterations. This implies that our algorithm has low computational costs.



(a) (b)

Fig. 2. Simulation results for problem (6) with objectives in (13) and (14).

V. Numerical Experiments

We use two experiments to illustrate the performance of our algorithm and compare it with other algorithms. Consider a network with $N = 36$ agents, and $\mathcal{G}$ varies from 9-cycle graph, 6×6 grid graph to Erdos-Renyi random graph with connectivity probability 0.4.

The first case is problem (6) with convex objectives. Here

$$f_i(x) = a_i e^{b_i x} + c_i e^{-d_i x}, \quad x \in X = [-3, 3], \tag{13}$$

where $a_i, c_i \sim \mathcal{U}(0, 1), \; b_i, d_i \sim \mathcal{U}(0, 0.2)$. We compare CPCA with Distributed Projected (sub)Gradient Descent (DPGD) in [16], whose vanishing step sizes are set based on [17], and weight matrix is set using Laplacian method [8]. Figure 2(a) shows the convergence results of both algorithms. The three curves corresponding to DPGD are too close to be distinguished from one another. Note that for our algorithm, the points on curves represent the number of inner consensus iterations needed to guarantee that the specified precision is achieved. We can see that DPGD has sublinear convergence rate, while the consensus iterations within our algorithm converge linearly.

The second case is problem (6) with non-convex Lipschitz objectives. Here

$$\begin{aligned} f_i(x) = a_i x^4 + b_i x^3 + c_i x^2 + d_i x + e_i, \\ x \in X = [-3, 3], \end{aligned} \tag{14}$$

where a_i to e_i satisfy normal distributions, with μ being $1/4, 2/3, -1/2, -2$ and 0 respectively, and σ all being 0.1. Then, $f(x)$ is roughly $\frac{1}{4}x^4 + \frac{2}{3}x^3 - \frac{1}{2}x^2 - 2x$, which is non-convex and has a global minima at $x = 1$. Figure 2(b) shows the convergence results of our algorithm. We can observe that in this case, CPCA can still yield ϵ-optimal solutions with $\mathcal{O}(\log(1/\epsilon))$ inner consensus iterations.

VI. Conclusion

In this paper, we propose CPCA for the distributed optimization problem with same local constraint sets and Lipschitz continuous univariate objective functions. CPCA is able to address the problem with non-convex objectives. This originates from the idea of solving an easier problem of optimizing the proxy for the global objective to obtain sufficiently precise approximate solutions. Also, CPCA has low computational complexities in that it is free from gradient or projection computations. This results from the scheme of employing average consensus to update coefficient vectors, not estimates of the optimizers. Future works include designing more efficient terminating rules for the iterations within the algorithm to reduce communication complexities, and developing similar algorithms for problems with multivariate objectives based on the idea of approximation.

References

[1] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[2] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, 2007.

[3] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links-Part I: Distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, 2007.

[4] J. He, P. Cheng, L. Shi, J. Chen, and Y. Sun, "Time synchronization in WSNs: A maximum-value-based consensus approach," *IEEE Transactions on Automatic Control*, vol. 59, no. 3, pp. 660–675, 2013.

[5] C. Zhao, J. He, P. Cheng, and J. Chen, "Consensus-based energy management in smart grid with transmission losses and directed communication," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2049–2061, 2016.

[6] J. He, C. Zhao, L. Cai, P. Cheng, and L. Shi, "Practical closed-loop dynamic pricing in smart grid for supply and demand balancing," *Automatica*, vol. 89, pp. 92–102, 2018.

[7] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[8] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[9] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[10] G. Qu and N. Li, "Accelerated distributed nesterov gradient descent," *IEEE Transactions on Automatic Control*, 2019.

[11] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual averaging for distributed optimization: Convergence analysis and network scaling," *IEEE Transactions on Automatic control*, vol. 57, no. 3, pp. 592–606, 2011.

[12] C. A. Uribe, S. Lee, A. Gasnikov, and A. Nedić, "A Dual Approach for Optimal Algorithms in Distributed Optimization over Networks," *arXiv e-prints*, p. arXiv:1809.00710, 2018.

[13] G. Lan, S. Lee, and Y. Zhou, "Communication-efficient algorithms for decentralized and stochastic optimization," *Mathematical Programming*, pp. 1–48, 2017.

[14] K. Scaman, F. Bach, S. Bubeck, L. Massoulié, and Y. T. Lee, "Optimal algorithms for non-smooth distributed optimization in networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 2740–2749.

[15] M. Zhu and S. Martínez, "On distributed convex optimization under inequality and equality constraints," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 151–164, 2011.

[16] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[17] S. Liu, Z. Qiu, and L. Xie, "Convergence rate analysis of distributed optimization with projected subgradient algorithm," *Automatica*, vol. 83, pp. 162–169, 2017.

[18] L. N. Trefethen, *Approximation theory and approximation practice*. SIAM, 2013, vol. 128.

[19] T. A. Driscoll, N. Hale, and L. N. Trefethen, "Chebfun guide," 2014.

[20] J. P. Boyd, *Solving Transcendental Equations: The Chebyshev Polynomial Proxy and Other Numerical Rootfinders, Perturbation Series, and Oracles*. SIAM, 2014, vol. 139.

[21] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis, "On distributed averaging algorithms and quantization effects," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2506–2517, 2009.

[22] A. Olshevsky, "Linear time average consensus and distributed optimization on fixed graphs," *SIAM Journal on Control and Optimization*, vol. 55, no. 6, pp. 3990–4014, 2017.