Consensus-based distributed streaming coupled tensor factorization

Liangliang Li*, Lin Gao*, Luigi Chisci§, Ping Wei*, Huaguo Zhang*, and Alfonso Farina‡
*School of Information and Communication Engineering
University of Electronic Science and Technology of China
Email: liliangliang_uestc@126.com, lingao_1014@126.com, {pwei, uestczhanghuaguo}@uestc.edu.cn
§Dipartimento di Ingegneria dell'Informazione (DINFO)
Università degli Studi di Firenze
Email: luigi.chisci@unifi.it
‡Selex ES (retired)
Email: alfonso.farina@outlook.it

***Abstract*—This paper discusses the problem of streaming coupled tensor factorization based on sensor networks, where each sensor observes only some features of the targets, and the measurements from sensors are provided in a streaming tensor fashion. Moreover, the observed features of different sensors might overlap (i.e., coupled tensor), and there is no central processing unit to collect all sensor data. Then, in our work, the *canonical polyadic* (CP) decomposition is exploited to perform local tensor decomposition based on the measurements of each sensor, and *average consensus* (AC) for diffusing information throughout the network. The proposed method is verified via simulations.**

***Index Terms*—Sensor network, radar network, streaming coupled tensor, CP decomposition, average consensus**

I. Introduction

Due to the ability of capturing multi-dimensional features, it is convenient to represent datasets of interest (e.g., hyperspectral super-resolution images [1] and array signals [2]) by coupled tensor models. In this paper, we are going to address the problem of streaming coupled tensor factorization, i.e., online tensor decomposition introduced by [3], [4], based on a *sensor network* (SN) [5], [6] which is made up of a set of synchronous sensor nodes (e.g., network of radars to improve the coverage and tracking accuracy of targets [7]). In such a case, nodes provide streaming measurements with respect to the time-varying tensor.

In order to capture the inherent dynamic characteristics of the tensor, it is more suitable to perform online processing instead of estimating a static tensor based on batch measurements [8]. Tensors [9] generalize matrices to more than two dimensions. Higher-order tensors and their decomposition problems have recently become pervasive in signal processing, data analytics, and machine learning [10], [11]. However, a majority of tensor decomposition algorithms have been designed for batch measurements, and relatively fewer works have been devoted to streaming measurements.

Seminal work on streaming tensor factorization (some literature uses the term tensor tracking) can be found in [3], where two adaptive *canonical polyadic* (CP) algorithms have been introduced for tracking 3-way streaming (i.e., 3-dimensional) tensors. Then, various streaming tensor factorization algorithms have been proposed [4], [12]–[18], and a comprehensive survey of the state-of-art is available in [19]. According to the taxonomy of [19], such algorithms can be classified into three types: streaming CP decompositions, Tucker decompositions, and streaming decompositions under other tensor formats. In [20], a generalized online CP tensor factorization and completion algorithm is proposed, and a remarkable feature of this work is that the estimated tensor before the last time can be further revised based on the current measurements. Then, the streaming *generalized CP* (GCP) decomposition method has been proposed in [21], which differs from traditional CP decompositions by introducing more types of optimization cost functions. Moreover, [22] has considered streaming tensor factorization based on parallel processing to speed up computation.

Nevertheless, the aforementioned methods are designed for single sensor processing. In practice, a SN can be deployed over the region of interest so as to get improved performance with respect to the single-sensor case. There are essentially two main approaches, i.e. centralized and distributed, for processing data from multiple sensors. The centralized approach makes use of a *fusion center* (FC) [23] to which all sensor nodes transmit their measured data. Conversely, the distributed approach [24] has no FC so that each node can exchange messages with neighbors only. In general, centralized processing clearly provides better performance, but at the cost of larger computational burden, lower fault tolerance and reduced scalability. In this paper, a distributed approach will therefore be considered.

Different from single sensor processing, under the scheme of the SN, we can model the observed data as coupled tensor model [25] (i.e., different measurements of the same targets). In particular, the aim of this work is to develop a fully distributed streaming coupled tensor factorization algorithm. To this end, we will first derive the streaming coupled tensor factorization algorithm based on the assumption that measurements of all sensors are available and cannot be processed in

TABLE I: Frequently used symbols in this paper

Notations	Descriptions
a	scalar
$\mathbf{a}$	vector
$\mathbf{A}$	matrix
$\boldsymbol{\mathcal{A}}$	tensor
$a_{i,j}$	(i,j)-th element of $\mathbf{A}$
$\mathcal{A}$	set, e.g., graph vertex set and edge set
$\mathbf{A}^\top$, $\mathbf{A}^{-1}$	transpose and inverse of $\mathbf{A}$
$\mathbf{U}^{(n)}$	n-th loading factor of $\boldsymbol{\mathcal{A}}$
$\mathbf{A}_{(n)}$	mode-n unfolding of $\boldsymbol{\mathcal{A}}$
$\circ$	outer product
$\odot$	Khatri-Rao product
$\times_n$	mode-n product
$\boldsymbol{\mathcal{A}}\prod_{n=1}^{N}\times_n\mathbf{U}^{(n)}$	$\boldsymbol{\mathcal{A}}\times_1\mathbf{U}^{(1)}\times_2\cdots\times_N\mathbf{U}^{(N)}$
$\bigodot_{n=1}^{N}\mathbf{U}^{(n)}$	$\mathbf{U}^{(N)}\odot\mathbf{U}^{(N-1)}\odot\cdots\odot\mathbf{U}^{(1)}$
$\mathbf{1}_N$	all-one vector with N elements
$\|\cdot\|$	Euclidean norm
$\lvert\cdot\rvert$	cardinality (i.e., number of elements)

a distributed manner. Then, we will exploit *average consensus* (AC) [26], [27] in order to spread the shared loading factors throughout the sensor network and thus carry out streaming tensor factorization in a fully distributed way. In the proposed algorithm, the tensor measurements at previous times are not needed, since the estimated parameters for each dimension at the current time are employed for initialization at the next time. In this way, computational complexity does not grow with time. Performance of the proposed algorithm is assessed via two simulation case-studies concerning loading factor and, respectively, *direction of arrival* (DOA) tracking problems. Our proposed method turns out to possess the following key features.

- It runs in a fully distributed manner, each node being able to keep its private dimensions.
- Only shared loading factors are exchanged via the SN, ensuring low communication complexity.

The rest of this paper is organized as follows. Section II introduces the necessary background, while the proposed method is presented in section III. Section IV provides simulation results to assess performance of the proposed method. Finally section V ends the paper. For the sake of convenience, used symbols are summarized in Table I. Moreover, definitions of Khatri-Rao and mode-n products in Table I can be found in [9, Table 2]; in particular, the Khatri-Rao product of $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_R] \in \mathbb{C}^{I\times R}$ and $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_R] \in \mathbb{C}^{J\times R}$ yields $\mathbf{C} \in \mathbb{C}^{I\times R}$ with columns $\mathbf{c}_r = \mathbf{a}_r \otimes \mathbf{b}_r$ where $\otimes$ denotes Kronecker product.

II. Background

A. Mathematical model of the SN

A crucial feature of distributed SNs is that no central processing unit is deployed, each node works in a peer-to-peer manner and communicates only with its neighboring nodes. A schematic diagram of distributed SN is shown in Fig. 1. Distributed sensor networks can be mathematically represented

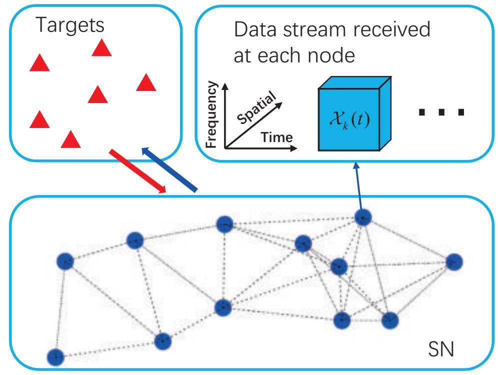


Fig. 1: Schematic diagram of distributed SN.

by graphs. To be more specific, let $(\mathcal{E}(t), \mathcal{V})$ denote an directed dynamic graph with vertex set $\mathcal{V} = \{1, 2, ..., K\}$ and possibly time-varying edge set $\mathcal{E}(t) = \{(k, k') \,|\, k, k' \in \mathcal{V}\}$ representing bi-directional data links. Let in-neighbors of node k be denoted by $\mathcal{N}_k(t) = \{k' \in \mathcal{V} \,|\, (k, k') \in \mathcal{E}(t)\}$ at time t, and $d_k(t) = |\mathcal{N}_k(t)|$ denote the degree (number of neighbors) of node k at time t.

B. Problem formulation

Denote a N-way tensor by $\boldsymbol{\mathcal{X}} \in \mathbb{C}^{I_1\times I_2\times\cdots\times I_N}$ and its entries by $x_{i_1,i_2,...,i_N}$, for $i_n = 1, \ldots, I_n$ and $n = 1, \ldots, N$. The rank-R CP decomposition of tensor $\boldsymbol{\mathcal{X}}$ is a sum of R rank-1 outer products (polyads), which can be expressed as

$$\boldsymbol{\mathcal{X}} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)} \triangleq [\![\mathbf{U}^{(n)}]\!]_{n=1}^{N}. \tag{1}$$

where $\mathbf{U}^{(n)} = [\mathbf{u}_1^{(n)}, ..., \mathbf{u}_R^{(n)}] \in \mathbb{C}^{I_n\times R}$ is the loading factor of mode n. Equivalently, $\boldsymbol{\mathcal{X}}$ can be expressed as a multilinear product with a diagonal core as follows

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{I}} \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}, \tag{2}$$

where $\boldsymbol{\mathcal{I}} \in \mathbb{C}^{R\times\cdots\times R}$ denotes the N-way unit diagonal tensor with $i_{r_1\ldots r_N} = 1$ if and only if $r_1 = \ldots = r_N$ and otherwise $i_{r_1\ldots r_N} = 0$. The multilinear product (i.e., mode-n product) of tensor $\boldsymbol{\mathcal{X}}$ is equivalent to the linear transform of tensor mode-n unfolding (see [9, Table 2])

$$\boldsymbol{\mathcal{X}} \times_n \mathbf{A} \xrightarrow[\text{unfolding}]{\text{mode}-n} \mathbf{A}\mathbf{X}_{(n)}, \tag{3}$$

where $\mathbf{A} \in \mathbb{C}^{I\times I_n}$, $\boldsymbol{\mathcal{X}} \times_n \mathbf{A} \in \mathbb{C}^{I_1\times\cdots\times I_{n-1}\times I\times I_{n+1}\times\cdots\times I_N}$ and the mode-n unfolding of $\boldsymbol{\mathcal{X}}$ is defined as

$$\mathbf{X}_{(n)} = \mathbf{U}^{(n)}\mathbf{M}_n^\top, \tag{4}$$

with $\mathbf{X}_{(n)} \in \mathbb{C}^{I_n\times(I_1\times\cdots\times I_{n-1}\times I_{n+1}\times\cdots\times I_N)}$ and $\mathbf{M}_n = \bigodot_{i=1,i\neq n}^{N}\mathbf{U}^{(i)}$. Specifically, a tensor $\boldsymbol{\mathcal{X}}$ can be uniquely obtained from the loading factors $\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(N)}$ (i.e., tensor decomposition is unique under mild conditions, which are different from matrix decomposition [28]).

In this paper, we consider the single-aspect streaming tensor model (see [19] for details). Let us define a N-way streaming tensor at time t and node $k \in \mathcal{V}$ as $\boldsymbol{\mathcal{Y}}_k(t) \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$, for $t = 1, 2, \ldots$. Specifically, at time t we have tensor measurements provided by node $k \in \mathcal{V}$ given by

$$\boldsymbol{\mathcal{Y}}_k(t) = \boldsymbol{\mathcal{I}} \prod_{n=1}^{N} \times_n \mathbf{U}_k^{(n)}(t) + \boldsymbol{\mathcal{N}}_k(t), \tag{5}$$

where $\boldsymbol{\mathcal{N}}_k(t)$ is the tensor measurement noise. Hereafter it is assumed that $\mathbf{U}_k^{(n)}(t) \in \mathbb{C}^{I_n \times R}$, $n = 1, 2, ..., N$, are time-varying loading factors evolving in time according to the following dynamic model

$$\mathbf{U}_k^{(n)}(t) = F\left(\mathbf{U}_k^{(n)}(t-1)\right) + \mathbf{N}_k^{(n)}(t), \tag{6}$$

where $F(\cdot)$ is the inherent dynamic model of the loading factors and $\mathbf{N}_k^{(n)}(t)$ the process noise at each node $k \in \mathcal{V}$. An example of definition of tensor $\boldsymbol{\mathcal{Y}}_k(t)$ relative to DOA tracking under known source number can be found in Section IV-B. Summarizing, the goal of this paper is to estimate, in an online manner, the loading factors $\mathbf{U}_k^{(n)}(t)$ at time t, for $n = 1, \ldots, N$, given the measured tensor $\boldsymbol{\mathcal{Y}}_k(t)$ for $k \in \mathcal{V}$.

III. Proposed method

Suppose there are P shared loading factors, and without loss of generality we assume that the first P loading factors of each node are shared. Here, we assume a hard coupling relationship between shared loading factors, i.e., $\mathbf{U}_k^{(p)}(t) = \mathbf{\Delta}^{(p)}(t)$ for $p = 1, \ldots, P$ and $k \in \mathcal{V}$ (see [29] for different types of coupling). Hence, the tensor slice received by sensor node k at each time turns out to be

$$\begin{aligned}\boldsymbol{\mathcal{Y}}_k(t) &= \left(\boldsymbol{\mathcal{I}} \prod_{p=1}^{P} \times_p \mathbf{U}_k^{(p)}(t)\right)\left(\prod_{q=P+1}^{N} \times_q \mathbf{U}_k^{(q)}(t)\right) + \boldsymbol{\mathcal{N}}_k(t) \\ &= [\![\mathbf{U}_k^{(n)}(t)]\!]_{n=1}^{N} + \boldsymbol{\mathcal{N}}_k(t),\end{aligned} \tag{7}$$

where $\mathbf{U}_k^{(q)}(t) \in \mathbb{C}^{I_q \times R}$ (for $q = P+1, \ldots, N$) are the private loading factors of node k. Then, the mode-n unfolding of the measured tensor is $\mathbf{Y}_{(n),k}(t) = \mathbf{U}_k^{(n)}(t)\mathbf{M}_{n,k}^{\top}(t)$ for any $k \in \mathcal{V}$.

If all sensor measurements are available, streaming coupled tensor decomposition can be expressed as the following optimization problem

$$\begin{aligned}\min_{\left\{\mathbf{U}_k^{(n)}(t)\right\}_{k\in\mathcal{V}, n\le N}} \; & \sum_{k=1}^{K} \omega_k \mathcal{L}_k\left(\boldsymbol{\mathcal{Y}}_k(t), [\![\mathbf{U}_k^{(n)}(t)]\!]_{n=1}^{N}\right) \\ & + \lambda_k \sum_{n=1}^{N} \mathcal{L}_k\left(\mathbf{U}_k^{(n)}(t), F\left(\mathbf{U}_k^{(n)}(t-1)\right)\right) \\ & \text{s.t.} \mathbf{U}_k^{(p)}(t) = \mathbf{\Delta}^{(p)}(t), k \in \mathcal{V}, p = 1, ..., P.\end{aligned} \tag{8}$$

where $\mathcal{L}_k(\cdot, \cdot)$ can be generic convex loss functions (e.g. squared Frobenius norm loss [30]) and ω_k, λ_k are suitably chosen weights. In practice, the dynamic model $F(\cdot)$ of the loading factor is complex and unknown. For simplicity, in this paper we only consider random-walk of the loading factors

$$\mathbf{U}_k^{(n)}(t) = \mathbf{U}_k^{(n)}(t-1) + \mathbf{N}_k^{(n)}(t). \tag{9}$$

Thus (8) can be reduced to

$$\begin{aligned}\min_{\left\{\mathbf{U}_k^{(n)}(t)\right\}_{k\in\mathcal{V}, n\le N}} \; & \sum_{k=1}^{K} \omega_k \mathcal{L}_k\left(\boldsymbol{\mathcal{Y}}_k(t), [\![\mathbf{U}_k^{(n)}(t)]\!]_{n=1}^{N}\right) \\ \text{s.t.} \quad & \mathbf{U}_k^{(p)}(t) = \mathbf{\Delta}^{(p)}(t), k \in \mathcal{V}, p = 1, ..., P.\end{aligned} \tag{10}$$

This problem is non-convex with respect to all arguments. However, if we fix mode n and assume that $\mathbf{U}_k^{(n')}(t)$ for $n' \neq n$ are known, (10) is converted into a convex optimization problem for $n = 1, \ldots, N$

$$\begin{aligned}\min_{\left\{\mathbf{U}_k^{(n)}(t)\right\}_{k\in\mathcal{V}}} \; & \sum_{k=1}^{K} \left[\omega_k \mathcal{L}_k\left(\boldsymbol{\mathcal{Y}}_k(t), [\![\mathbf{U}_k^{(n')}(t)]\!]_{n'=1}^{N}\right)\right] \\ \text{s.t.} \quad & \mathbf{U}_k^{(p)}(t) = \mathbf{\Delta}^{(p)}(t),\, k \in \mathcal{V}, p = 1, ..., P,\end{aligned} \tag{11}$$

which can be solved with the *alternating direction method of multipliers* (ADMM) [31], wherein the augmented Lagrangian is given by

$$\begin{aligned}& L\left(\mathbf{U}_k^{(p)}(t), \mathbf{\Delta}^{(p)}(t), \boldsymbol{\mu}_k^{(n)}(t)\right) \\ &= \sum_{k=1}^{K} \left[\omega_k \mathcal{L}_k\left(\boldsymbol{\mathcal{Y}}_k(t), [\![\mathbf{U}_k^{(n)}(t)]\!]_{n=1}^{N}\right)\right] \\ &\quad + \frac{\rho}{2}\left\|\mathbf{U}_k^{(p)}(t) - \mathbf{\Delta}^{(p)}(t) + \boldsymbol{\mu}_k^{(p)}(t)\right\|_F^2,\end{aligned} \tag{12}$$

where ρ is a regularization factor and $\boldsymbol{\mu}_k^{(n)}(t)$ a dual variable. In the case of squared Frobenius norm loss, after the initialization of $\mathbf{U}_{k,0}^{(n)}(t), \mathbf{\Delta}_0^{(n)}(t), \boldsymbol{\mu}_{k,0}^{(n)}(t)$ (for $k \in \mathcal{V}$ and $n = 1, ..., N$), the update of the shared loading factor at j-th iteration is given, for $j = 1, 2, \ldots$, by

$$\begin{aligned}\mathbf{U}_{k,j}^{(p)}(t) =& \min_{\mathbf{X}} \omega_k \left\|\mathbf{Y}_{(p),k}(t) - \mathbf{X}\mathbf{M}_{p,k,j-1}^{\top}(t)\right\|_F^2 \\ & + \frac{\rho}{2}\left\|\mathbf{X} - \mathbf{\Delta}_{k,j-1}^{(p)}(t) + \boldsymbol{\mu}_{k,j-1}^{(p)}(t)\right\|_F^2,\end{aligned} \tag{13}$$

where $\mathbf{M}_{p,k,j}(t) = \bigodot_{i=1, i\neq n}^{N} \mathbf{U}_{k,j}^{(i)}(t)$. Then, the p-th shared loading factor (for $p = 1, \ldots, P$) is updated according to

$$\mathbf{\Delta}_j^{(p)}(t) = \frac{1}{K}\sum_{k=1}^{K}\left(\mathbf{U}_{k,j}^{(p)}(t) + \boldsymbol{\mu}_{k,j-1}^{(p)}(t)\right), \tag{14}$$

where $\boldsymbol{\mu}_{k,j}^{(p)}(t) = \boldsymbol{\mu}_{k,j-1}^{(p)}(t) + \mathbf{U}_{k,j}^{(p)}(t) - \mathbf{\Delta}_j^{(p)}(t)$. The iterative optimization can be stopped according to a convergence condition, i.e., if the estimated shared loading factors at the current iteration are close to the results at the previous iteration.

The averaging over nodes k in (14) suggests resorting to AC so as to perform fully distributed estimation without any FC. In the proposed distributed streaming coupled tensor decomposition method, we first use the *alternating least squares* (ALS) algorithm [32] to estimate the loading factors of each node and then use AC to improve the accuracy of the shared loading factors between nodes. In particular, as far as the tensor slice of each node $k \in \mathcal{V}$ is acquired, from (13) the loading factors estimated at j-th iteration turn out to be given (for $p = 1, \dots, P$ and $q = P+1, \dots, N$) by

$$\begin{aligned}\widehat{\mathbf{U}}_{k,j}^{(p)}(t) &= \left[\omega_k \mathbf{Y}_{(p),k}(t)\mathbf{M}_{p,k,j-1}(t-1) + \frac{\rho}{2}\mathbf{Q}_{p,k,j-1}(t)\right] \\ &\quad \times \left[\omega_k \mathbf{P}_{p,k,j-1}(t) + \frac{\rho}{2}\mathbf{I}_R\right]^{-1}, \end{aligned} \tag{15}$$

$$\widehat{\mathbf{U}}_{k,j}^{(q)}(t) = \mathbf{Y}_{(q),k}(t)\left[\mathbf{M}_{q,k,j-1}(t-1)\left(\mathbf{P}_{q,k,j-1}(t)\right)^{-1}\right], \tag{16}$$

where $\mathbf{Q}_{p,k,j}(t) = \widehat{\mathbf{U}}_{k,j}^{(p)}(t-1) - \boldsymbol{\mu}_{k,j}^{(p)}(t-1)$, $\mathbf{P}_{n,k,j}(t) = \mathbf{M}_{n,k,j}^{\top}(t-1)\mathbf{M}_{n,k,j}(t-1)$. It is worth pointing out that (15) and (16) are computed independently at each node $k \in \mathcal{V}$, and this low-rank tensor decomposition is essentially unique under mild conditions with the following permutation and scaling ambiguity

$$\widehat{\mathbf{U}}_k^{(n)}(t) = \mathbf{U}_k^{(n)}(t)\boldsymbol{\Pi}_k(t)\boldsymbol{\Lambda}_k^{(n)}(t), \tag{17}$$

where $\boldsymbol{\Pi}_k(t)$ is a permutation matrix and $\boldsymbol{\Lambda}_k^{(n)}(t)$ are full-rank diagonal scaling matrices such that $\boldsymbol{\Lambda}^{(1)}(t)\cdots\boldsymbol{\Lambda}^{(N)}(t) = \mathbf{I}$. Specifically, $\boldsymbol{\Pi}_k(t)$ is used to ensure consistency of elements of each dimension between real and estimated tensors, which can be obtained via an assignment algorithm like, e.g., the Hungarian algorithm [33].

In order to aggregate the information provided by the tensor slices of all nodes, we propose to compute (14) via AC. In particular, we first perform CP decomposition from (15) and (16) to achieve $\widehat{\mathbf{U}}_k^{(n)}(t)$ solely based on the received tensor slice of each node $k \in \mathcal{V}$, then AC is carried out for shared loading factors at each node via the following steps.

- Define $\overline{\mathbf{U}}_{k,0}^{(p)}(t) = \widehat{\mathbf{U}}_{k,j}^{(p)}(t)$, $\overline{\boldsymbol{\mu}}_{k,0}^{(p)}(t) = \boldsymbol{\mu}_{k,j-1}^{(p)}(t)$
- For $\ell = 1, \dots, L$ compute

$$\begin{aligned}\overline{\mathbf{U}}_{k,\ell}^{(p)}(t) &= w_{k,k}(t)\overline{\mathbf{U}}_{k,\ell-1}^{(p)}(t) + \sum_{k' \in \mathcal{N}_k(t)} w_{k,k'}(t)\overline{\mathbf{U}}_{k',\ell-1}^{(p)}(t), \\ \overline{\boldsymbol{\mu}}_{k,\ell}^{(p)}(t) &= w_{k,k}(t)\overline{\boldsymbol{\mu}}_{k,\ell-1}^{(p)}(t) + \sum_{k' \in \mathcal{N}_k(t)} w_{k,k'}(t)\overline{\boldsymbol{\mu}}_{k',\ell-1}^{(p)}(t),\end{aligned} \tag{18}$$

  where $w_{k,k'}(t)$ are the consensus weights of node k, whose definition can be found in IV-A.
- As far as L iterations are accomplished, we set

$$\begin{aligned}\widehat{\mathbf{U}}_{k,j}^{(p)}(t) &= \overline{\mathbf{U}}_{k,L}^{(p)}(t) + \overline{\boldsymbol{\mu}}_{k,L}^{(p)}(t), \\ \boldsymbol{\mu}_{k,j}^{(p)}(t) &= \overline{\boldsymbol{\mu}}_{k,L}^{(p)}(t) + \overline{\mathbf{U}}_{k,0}^{(p)}(t) - \widehat{\mathbf{U}}_k^{(p)}(t).\end{aligned} \tag{19}$$

Then, we use the outcome of consensus as the initial value of the ALS algorithm at time $t+1$. The proposed method is summarized in Algorithm 1.

Algorithm 1: Proposed method (time t, node $k \in \mathcal{V}$)

Input: Tensor slices $\boldsymbol{\mathcal{Y}}_k(t)$; number of consensus steps L; AC weighting matrix $\mathbf{W}(t)$; ADMM convergence threshold η_k; $\{\widehat{\mathbf{U}}_k^{(n)}(t-1)\}_{n=1}^N$ and $\boldsymbol{\mu}_k^{(p)}(t-1)$ for $n = 1, \dots, N$ and $p = 1, \dots, P$.

Set $repeat\_mark = 1$, set $j = 0$;
Let $\{\widehat{\mathbf{U}}_{k,0}^{(n)}(t)\}_{n=1}^N = \{\widehat{\mathbf{U}}_k^{(n)}(t-1)\}_{n=1}^N$;
Let $\boldsymbol{\mu}_{k,0}^{(p)}(t) = \boldsymbol{\mu}_k^{(p)}(t-1)$;
while *(*repeat_mark*)* **do**
  $j = j + 1$;
  Step 1: Alternating least square
  for $p = 1, \dots, P$ **do**
    Carry out (15) to obtain $\widehat{\mathbf{U}}_{k,j}^{(p)}(t)$;
  end
  for $q = P+1, \dots, N$ **do**
    Carry out (16) to obtain $\widehat{\mathbf{U}}_{k,j}^{(q)}(t)$;
  end
  Step 2: Average consensus
  for $p = 1, ..., P$ **do**
    Set $\overline{\mathbf{U}}_{k,0}^{(p)}(t) = \widehat{\mathbf{U}}_{k,j}^{(p)}(t)$, $\overline{\boldsymbol{\mu}}_{k,0}^{(p)}(t) = \boldsymbol{\mu}_{k,j-1}^{(p)}(t)$;
    for $\ell = 1, ..., L$ **do**
      Broadcast $\overline{\mathbf{U}}_{k,\ell-1}^{(p)}(t)$, $\overline{\boldsymbol{\mu}}_{k,\ell-1}^{(p)}(t)$ and receive $\overline{\mathbf{U}}_{k',\ell-1}^{(p)}(t)$, $\overline{\boldsymbol{\mu}}_{k',\ell-1}^{(p)}(t)$, for $k' \in \mathcal{N}_k$;
      Carry out (18) to obtain $\overline{\mathbf{U}}_{k,\ell}^{(p)}(t)$ and $\overline{\boldsymbol{\mu}}_{k,\ell}^{(p)}(t)$;
    end
    Compute $\widehat{\mathbf{U}}_{k,j}^{(p)}(t)$ and $\boldsymbol{\mu}_{k,j}^{(p)}(t)$ with (19);
  end
  if $||\widehat{\mathbf{U}}_{k,j}^{(n)}(t) - \widehat{\mathbf{U}}_{k,j-1}^{(n)}(t)||_F^2 \leq \eta_k$ *(for all* $n = 1, \dots, N$*)* **then**
    Set $repeat\_mark = 0$;
  end
end
Set $\widehat{\mathbf{U}}_k^{(n)}(t) = \widehat{\mathbf{U}}_{k,j}^{(n)}(t)$, $n = 1, \dots, N$.
Output: Estimated loading factors $\widehat{\mathbf{U}}_k^{(n)}(t)$ at node k.

IV. Simulation Examples

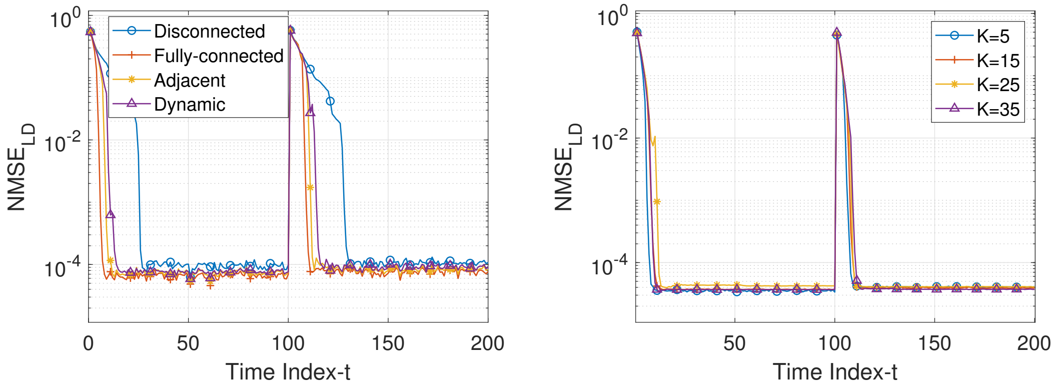
In this section, the performance of the proposed method is verified via two simulation case-studies concerning loading factor estimation and DOA tracking problems.

A. Loading factor estimation case-study

Suppose that: (1) the entries of $\mathbf{U}_k^{(n)}(0)$ follow an i.i.d Gaussian distribution with zero mean and unit variance; (2) the entries of measurement noise $\boldsymbol{\mathcal{N}}_k$ follow an i.i.d Gaussian distribution with zero mean and variance σ^2; (3) the elements of the process noise loading factor $\mathbf{N}_k(t)$ are i.i.d. Gaussian with zero mean and variance ϵ^2 (i.e., time-varying factor). To evaluate estimation accuracy, we use the *normalized mean square error* (NMSE) of loading factors defined as follows.

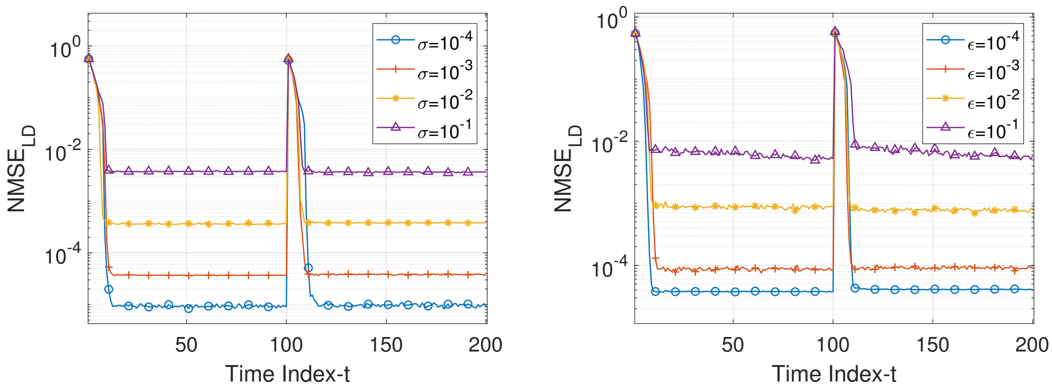
$$\mathrm{NMSE}_{\mathrm{LD}}(t) = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1}^{N}\frac{||\widehat{\mathbf{U}}_k^{(n)}(t) - \mathbf{U}_k^{(n)}(t)||}{||\mathbf{U}_k^{(n)}(t)||}. \tag{20}$$

We consider the following network types and the corre-



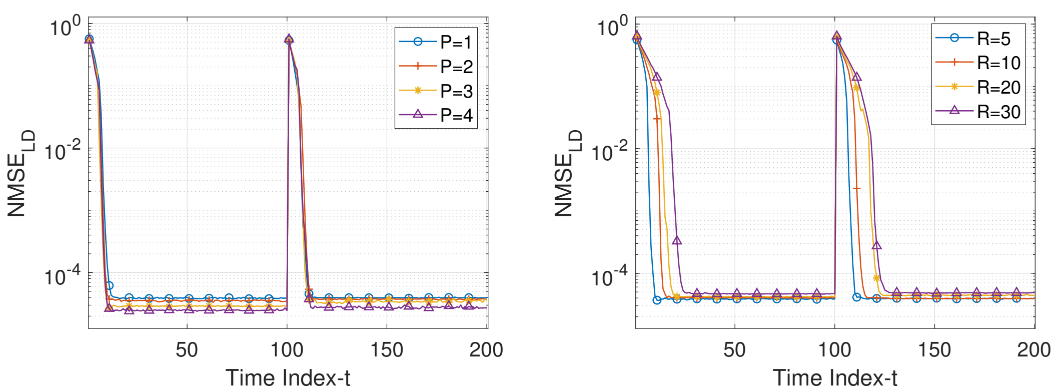
(a) (b)

Fig. 2: NMSE of estimated loading factors for different network types (a) and node number K (b).



(a) (b)

Fig. 3: NMSE of estimated loading factors for different noise power (a) and time-varying factor (b).



(a) (b)

Fig. 4: NMSE of estimated loading factors (a) vs. number P of public loading factors; (b) vs. CP rank R.

sponding AC weighting matrix W [26]:

- disconnected network, i.e. all nodes have no neighbors, with $\mathbf{W} = \mathbf{I}_K$;
- fully-connected network, i.e. all nodes are connected to each other, with $\mathbf{W} = \mathbf{1}_K \mathbf{1}_K^\top / K$;
- adjacent network, i.e., each node is connected to at most two neighboring nodes, with

$$\mathbf{W} = \begin{bmatrix} 1/2 & 1/2 & 0 & 0 & \cdots & 0 \\ 0 & 1/3 & 1/3 & 1/3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 1/2 \end{bmatrix};$$

- dynamic network, i.e. the network topology changes at each time, with Metropolis consensus weights defined as [26]

$$w_{k,k'}(t) = \begin{cases} \dfrac{1}{1+\max\{d_k(t), d_{k'}(t)\}} & (k,k') \in \mathcal{E}(t), \\ 1 - \displaystyle\sum_{\{k,k'\}\in\mathcal{E}(t)} w_{k,k'}(t) & k = k', \\ 0 & \text{otherwise.} \end{cases}$$

In our simulations, we set $N = 4$ and $\boldsymbol{\mathcal{Y}}_k(t) \in \mathbb{C}^{10\times10\times10\times10}$. Besides, a significant change in loading factors is assumed at $t = 100$ (i.e. re-initialize loading factors), and all the results are averaged over 1000 Monte Carlo trials. Weighting, regularization and stopping threshold parameters are set to $\omega_k = 1, \rho = 1$ and $\eta_k = 10^{-6}$ for any $k \in \mathcal{V}$. Specifically, at $t = 0$ we initialize $\widehat{\mathbf{U}}_k^{(n)}(0)$ and $\boldsymbol{\mu}_k^{(p)}(0)$ randomly, with all elements of $\widehat{\mathbf{U}}_k^{(n)}(0)$ and $\boldsymbol{\mu}_k^{(p)}(0)$ i.i.d. Gaussian with zero-mean and unit-variance. Specifically, for the sake of simplicity, we do not consider the limited resolution of sensors and assume detection probability $P_d = 1$. Simulation results for different network types and number of nodes are shown in Figs. 2a and 2b. The performance for different measurement noise and process noise variances σ^2 and, respectively, ϵ^2 are shown in Figs. 3a and 3b. In addition, performance for different number P of shared loading factors and CP rank R are shown in Figs. 4a and 4b. It can be seen from the aforementioned results that the proposed algorithm has good convergence and that the parameter changes (i.e. σ, ϵ, P, R) do not significantly affect loading factor estimation performance, thus verifying the effectiveness of the proposed method.

B. DOA tracking case

In this subsection, we consider a networked *multiple-input multiple-output* (MIMO) radar [34] scenario for DOA tracking of moving targets. The MIMO radar employs multiple antennas, at both the transmitter and the receiver sites, which transmit linearly independent waveforms to offer superior capabilities for target detection and tracking. In practice, reaching independent waveforms is a hard job. In our simulations, we assume linearly independent waveforms and that the received signals have been synchronized. Assuming a transmitter array and multiple receiver arrays, the matched filtered signal from each receiver array in a *coherent processing interval* (CPI) can be expressed as the following tensor [35]

$$\boldsymbol{\mathcal{Y}}_k(t_n) = \boldsymbol{\mathcal{I}} \times_1 \mathbf{A}(t_n) \times_2 \mathbf{B}(t_n) \times_3 \mathbf{C}(t_n) + \boldsymbol{\mathcal{N}}_k(t_n), \quad (21)$$

where the Vandermonde matrices $\mathbf{A}(t_n) = [\mathbf{a}(\phi_1(t_n)), ..., \mathbf{a}(\phi_K(t_n))] \in \mathbb{C}^{M_T\times R}$ and $\mathbf{B}(t_n) = [\mathbf{b}(\theta_1(t_n)), ..., \mathbf{b}(\theta_K(t_n))] \in \mathbb{C}^{M_R\times R}$ represent the transmit and receive steering matrices with K, M_T, M_R corresponding to numbers of targets, transmitting antennas and receiving antennas respectively. The factor $\mathbf{C}(t_n) \in \mathbb{C}^{F\times R}$ is the reflection coefficient matrix including the Doppler effect and fading of target *radar cross section* (RCS), where F is pulse number during a CPI. We set $R = 10, M_T = 4, M_R = 4, F = 10, \sigma = 0.001$ in our simulations. The simulation scenario is shown in Fig. 5. Moreover, we assume that the transmitter is far away from the receiver and in the simulation the transmitter is located at $(400km, 0)$. The results averaged over 1000 Monte Carlo trials are displayed in Fig. 6. From the simulation results, when the algorithm converges, the average estimation error of DOA is less than $0.1°$.

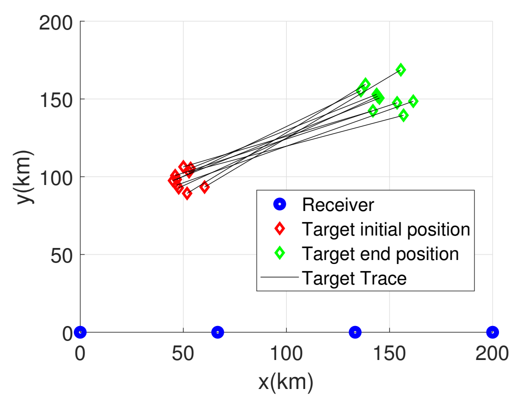


Fig. 5: DOA tracking scenario.

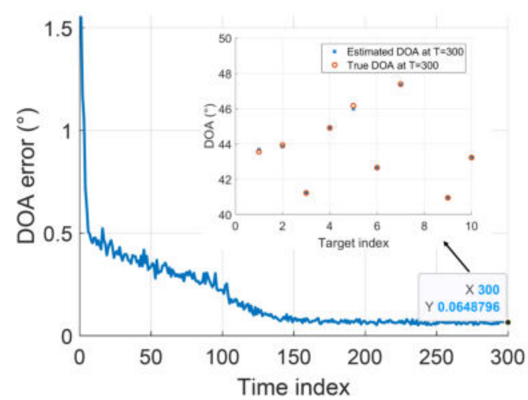


Fig. 6: DOA tracking error.

V. Conclusion

This paper has addressed distributed streaming coupled tensor decomposition via sensor networks. A key feature of the proposed algorithm is the adoption of average consensus for diffusing the shared loading factors of the nodes throughout the network. Simulation results have demonstrated the effectiveness of the proposed algorithm. Possible future work will concern: 1) extension to time-varying rank of tensors; 2) generalization of the dynamic model $F(\cdot)$ for loading factors.

References

[1] C. I. Kanatsoulis, X. Fu, N. D. Sidiropoulos, and W.-K. Ma, "Hyperspectral super-resolution: A coupled tensor factorization approach," *IEEE Transactions on Signal Processing*, vol. 66, no. 24, pp. 6503–6517, 2018.

[2] H. Zheng, Z. Shi, C. Zhou, M. Haardt, and J. Chen, "Coupled coarray tensor CPD for DOA estimation with coprime L-shaped array," *IEEE Signal Processing Letters*, vol. 28, pp. 1545–1549, 2021.

[3] D. Nion and N. D. Sidiropoulos, "Adaptive algorithms to track the PARAFAC decomposition of a third-order tensor," *IEEE Transactions on Signal Processing*, vol. 57, no. 6, pp. 2299–2310, 2009.

[4] K. Abed-Meraim, N. L. Trung, A. Hafiane, *et al.*, "Robust tensor tracking with missing data and outliers: Novel adaptive CP decomposition and convergence analysis," *IEEE Transactions on Signal Processing*, vol. 70, pp. 4305–4320, 2022.

[5] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[6] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras, "Applications of wireless sensor networks: an up-to-date survey," *Applied system innovation*, vol. 3, no. 1, p. 14, 2020.

[7] S. H. Javadi and A. Farina, "Radar networks: A review of features and challenges," *Information Fusion*, vol. 61, pp. 48–55, 2020.

[8] T. Kolajo, O. Daramola, and A. Adebiyi, "Big data stream analysis: a systematic literature review," *Journal of Big Data*, vol. 6, no. 1, p. 47, 2019.

[9] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[10] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[11] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[12] W. Hu, X. Li, X. Zhang, X. Shi, S. Maybank, and Z. Zhang, "Incremental tensor subspace learning and its applications to foreground segmentation and tracking," *International Journal of Computer Vision*, vol. 91, pp. 303–327, 2011.

[13] M. Mardani, G. Mateos, and G. B. Giannakis, "Subspace learning and imputation for streaming big data matrices and tensors," *IEEE Transactions on Signal Processing*, vol. 63, no. 10, pp. 2663–2677, 2015.

[14] H. Kasai, "Online low-rank tensor subspace tracking from incomplete data by CP decomposition using recursive least squares," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2519–2523, IEEE, 2016.

[15] Y. Du, Y. Zheng, K.-c. Lee, and S. Zhe, "Probabilistic streaming tensor decomposition," in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 99–108, IEEE, 2018.

[16] H. Kasai, "Fast online low-rank tensor subspace tracking by CP decomposition using recursive least squares from incomplete observations," *Neurocomputing*, vol. 347, pp. 177–190, 2019.

[17] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, "Streaming tensor factorization for infinite data sources," in *Proceedings of the 2018 SIAM International Conference on Data Mining*, pp. 81–89, SIAM, 2018.

[18] S. Zhou, N. X. Vinh, J. Bailey, Y. Jia, and I. Davidson, "Accelerating online CP decompositions for higher order tensors," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1375–1384, 2016.

[19] K. Abed-Meraim, N. L. Trung, A. Hafiane, *et al.*, "A contemporary and comprehensive survey on streaming tensor decomposition," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[20] C. Yang, C. Qian, and J. Sun, "Gocpt: Generalized online canonical polyadic tensor factorization and completion," *arXiv preprint arXiv:2205.03749*, 2022.

[21] E. T. Phipps, N. T. Johnson, and T. G. Kolda, "Streaming generalized canonical polyadic tensor decompositions," in *Proceedings of the Platform for Advanced Scientific Computing Conference*, pp. 1–10, 2023.

[22] X. Wang, L. T. Yang, X. Chen, M. J. Deen, and J. Jin, "Improved multi-order distributed HOSVD with its incremental computing for smart city services," *IEEE Transactions on Sustainable Computing*, vol. 6, no. 3, pp. 456–468, 2018.

[23] S. He, H.-S. Shin, S. Xu, and A. Tsourdos, "Distributed estimation over a low-cost sensor network: A review of state-of-the-art," *Information Fusion*, vol. 54, pp. 21–43, 2020.

[24] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

[25] C. Chatzichristos, S. Van Eyndhoven, E. Kofidis, and S. Van Huffel, "Coupled tensor decompositions for data fusion," in *Tensors for data processing*, pp. 341–370, Elsevier, 2022.

[26] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pp. 63–70, IEEE, 2005.

[27] G. Battistelli, L. Chisci, G. Mugnai, A. Farina, and A. Graziano, "Consensus-based linear and nonlinear filtering," *IEEE Transactions on Automatic Control*, vol. 60, no. 5, pp. 1410–1415, 2014.

[28] I. Domanov and L. De Lathauwer, "On the uniqueness of the canonical polyadic decomposition of third-order tensors—part I: Basic results and uniqueness of one factor matrix," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 855–875, 2013.

[29] C. Schenker, J. E. Cohen, and E. Acar, "A flexible optimization framework for regularized matrix-tensor factorizations with linear couplings," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 3, pp. 506–521, 2020.

[30] M. Gavish and D. L. Donoho, "Optimal shrinkage of singular values," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2137–2152, 2017.

[31] 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.

[32] R. A. Harshman *et al.*, "Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis," 1970.

[33] R. Jonker and T. Volgenant, "Improving the Hungarian assignment algorithm," *Operations research letters*, vol. 5, no. 4, pp. 171–175, 1986.

[34] A. M. Haimovich, R. S. Blum, and L. J. Cimini, "MIMO radar with widely separated antennas," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 116–129, 2007.

[35] J. Du, J. Dong, L. Jin, and F. Gao, "Bayesian robust tensor factorization for angle estimation in bistatic MIMO radar with unknown spatially colored noise," *IEEE Transactions on Signal Processing*, vol. 70, pp. 6051–6064, 2022.