# Dynamic L1-Norm Tucker Tensor Decomposition

Dimitris G. Chachlakis, *Member, IEEE*, Mayur Dhanaraj, *Student Member, IEEE*, Ashley Prater-Bennette, *Member, IEEE*, and Panos P. Markopoulos, *Member, IEEE*

***Abstract*—Tucker decomposition is a standard method for processing multi-way (tensor) measurements and finds many applications in machine learning and data mining, among other fields. When tensor measurements arrive in a streaming fashion or are too many to jointly decompose, incremental Tucker analysis is preferred. In addition, dynamic adaptation of bases is desired when the nominal data subspaces change. At the same time, it has been documented that outliers in the data can significantly compromise the performance of existing methods for dynamic Tucker analysis. In this work, we present Dynamic L1-Tucker: an algorithm for dynamic and outlier-resistant Tucker analysis of tensor data. Our experimental studies on both real and synthetic datasets corroborate that the proposed method (i) attains high bases estimation performance, (ii) identifies/rejects outliers, and (iii) adapts to changes of the nominal subspaces.**

***Index Terms*—Data analysis, L1-norm, outliers, robust, tensors, tucker decomposition.**

## I. Introduction

DATA collections across diverse sensing modalities are naturally stored and processed in the form of $N$-way arrays, also known as *tensors* [1]. Tucker decomposition is a standard method for tensor analysis, with important applications in machine learning [2]–[4], pattern recognition [5], [6], communications [7]–[9], and computer vision [10]–[12], among other fields. Common uses of Tucker include compression, denoising, and model identification. Canonical Polyadic Decomposition (CPD) [13], [14], also known as Parallel Factor Analysis (PARAFAC), is another successful tensor analysis scheme with many applications in data mining and machine learning.

Tucker can be viewed as a high-order extension of Principal-Component Analysis (PCA) [15]. Similar to PCA, which jointly analyzes a collection of vectors, Tucker analyzes a collection of $(N \geq 1)$-way tensors to extract one orthonormal basis for each tensor mode. Instead of applying PCA on vectorized measurements, Tucker treats multi-way measurements in their tensor form, thus leveraging inherent data structure and allowing for superior inference. Standard solvers for Tucker are the Higher-Order Singular Value Decomposition (HOSVD) and the Higher-Order Orthogonal Iterations (HOOI) [16], [17].

The merits of Tucker analysis have been demonstrated in a wide range of applications. However, it is well-documented that Tucker is very sensitive against faulty measurements (outliers). Outliers appear often in modern datasets due to sensor malfunctions, errors in data storage/transfer, and even deliberate dataset contamination in adversarial environments [18]–[20]. The outlier-sensitivity of Tucker is attributed to its L2-norm (Frobenius) formulation, which places quadratic emphasis to peripheral tensor entries. To remedy the impact of outliers, researchers have proposed robust reformulations of Tucker. For instance, Higher-Order Robust PCA (HoRPCA) [21] models and decomposes the processed tensor as the sum of a low multilinear rank tensor (nominal data) and a sparse tensor (outliers). Another straightforward robust reformulation is L1-Tucker [22], [23], which derives by simple substitution of the L2-norm in Tucker by the more robust L1-norm (not to be confused with sparsity-inducing L1-norm regularization schemes). Algorithms for the (approximate) solution of L1-Tucker have been proposed in [22]–[28].

In many applications of interest, the tensor measurements arrive in a streaming way. Accordingly, the sought-after Tucker bases have to be computed incrementally. Incremental solvers are also preferred, from a computational standpoint, when there are too many collected measurements to efficiently process them as a batch. For such cases, researchers have proposed an array of algorithms for incremental Tucker decomposition, including Dynamic Tensor Analysis (DTA), Streaming Tensor Analysis (STA), Window-based Tensor Analysis (WTA) [29], [30], and Accelerated Online Low-Rank Tensor Learning (ALTO) [31], to name a few. Despite their computational merits, similar to batch Tucker analysis, most of the existing incremental methods are sensitive against outliers.

In this work, we present Dynamic L1-Tucker: a scalable method for incremental L1-Tucker analysis, with the ability to (i) provide quality estimates of the Tucker bases, (ii) detect and reject outliers, and (iii) adapt to changes of the nominal subspaces. The rest of this paper is organized as follows. In Section II, we introduce notation and provide an overview of the relevant technical background (tensors, Tucker decomposition, L1-Tucker, and existing methods for dynamic/incremental Tucker). In Section III, we formally state the problem of interest. In Section IV, we present the proposed Dynamic L1-Tucker

Manuscript received August 2, 2020; revised November 20, 2020 and January 19, 2021; accepted January 25, 2021. Date of publication February 18, 2021; date of current version March 29, 2021. This material is based upon work supported in part by the National Science Foundation under Award OAC-1808582, and in part by the Air Force Office of Scientific Research under Awards FA9550-20-1-0039 and 18RICOR029. Cleared for public release (case number AFRL-2021-0085). The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Hongyang Chen. *(Corresponding author: Panos Markopoulos.)*

Dimitris G. Chachlakis, Mayur Dhanaraj, and Panos P. Markopoulos are with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: dimitris@mail.rit.edu; mxd6023@rit.edu; panos@rit.edu).

Ashley Prater-Bennette is with the Air Force Research Laboratory, Information Directorate, Rome, NY 13441 USA (e-mail: ashley.prater-bennette@us.af.mil).

Digital Object Identifier 10.1109/JSTSP.2021.3058846

(D-L1-Tucker) method. Section V holds extensive experimental studies on synthetic and real datasets. Concluding remarks are drawn in Section VI.

## II. Technical Background

### A. Notation and Tensor Preliminaries

In this manuscript, vectors and matrices are denoted by lower- and upper-case bold letters, respectively –e.g., $\mathbf{x} \in \mathbb{R}^{D_1}$ and $\mathbf{X} \in \mathbb{R}^{D_1 \times D_2}$. $N$-way tensors are denoted by upper-case calligraphic bold letters –e.g., $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{D_1 \times \ldots \times D_N}$. Collections/sets of tensors are denoted by upper-case calligraphic letters –e.g., $\mathcal{X} = \{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}\}$. The squared Frobenius/L2 norm, $\|\cdot\|_F^2$, returns the sum of squared entries of its tensor argument while the L1-norm, $\|\cdot\|_1$, returns the sum of the absolute entries of its tensor argument. $\mathbb{S}_{D\times d} = \{\mathbf{Q} \in \mathbb{R}^{D\times d} : \mathbf{Q}^\top \mathbf{Q} = \mathbf{I}_d\}$ is the Stiefel manifold of rank-$d$ orthonormal matrices in $\mathbb{R}^D$. Each entry of $\boldsymbol{\mathcal{X}}$ is identified by $N$ indices $\{i_n\}_{n=1}^N$ such that $i_n \leq D_n$ for every $n \in [N] = \{1, 2, \ldots, N\}$. For every $n \in [N]$, $\boldsymbol{\mathcal{X}}$ can be seen as a collection of $P_n = \prod_{m\in[N]\setminus n} D_m$ length-$D_n$ vectors known as mode-$n$ fibers of $\boldsymbol{\mathcal{X}}$. For instance, given a fixed set of indices $i_{m\in[N]\setminus n}$, $\boldsymbol{\mathcal{X}}(i_1, \ldots, i_{n-1}, :, i_{n+1}, \ldots, i_N)$ is a mode-$n$ fiber of $\boldsymbol{\mathcal{X}}$. A matrix that has as columns all the mode-$n$ fibers of $\boldsymbol{\mathcal{X}}$ is called the mode-$n$ unfolding (or, flattening) of $\boldsymbol{\mathcal{X}}$ and will henceforth be denoted as $\texttt{mat}(\boldsymbol{\mathcal{X}}, n) \in \mathbb{R}^{D_n \times P_n}$. $\boldsymbol{\mathcal{X}} \times_n \mathbf{A}$ is the mode-$n$ product of tensor $\boldsymbol{\mathcal{X}}$ with matrix $\mathbf{A}$ of conformable size and $\boldsymbol{\mathcal{X}} \times_{n\in[N]} \mathbf{Q}_n^\top$ compactly denotes the multi-way product $\boldsymbol{\mathcal{X}} \times_1 \mathbf{Q}_1^\top \times_2 \mathbf{Q}_2^\top \ldots \times_N \mathbf{Q}_N^\top$. In accordance with the common convention, the order in which the mode-$n$ fibers of $\boldsymbol{\mathcal{X}}$ appear in $\texttt{mat}(\boldsymbol{\mathcal{X}}, n)$ is as specified in [16].

### B. Tucker Decomposition

Consider coherent tensor measurements $\boldsymbol{\mathcal{X}}_t \in \mathbb{R}^{D_1 \times \ldots \times D_N}$, $t = 1, 2, \ldots, T$. Also, define their concatenation tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{D_1 \times \ldots \times D_N \times T}$, such that $\boldsymbol{\mathcal{X}}(:, :, \ldots, :, t) = \boldsymbol{\mathcal{X}}_t$. Tucker analysis of the measurement batch $\{\boldsymbol{\mathcal{X}}_t\}_{t=1}^T$ is formulated as

$$\max_{\{\mathbf{Q}_n \in \mathbb{S}_{D_n \times d_n}\}_{n\in[N]}} \left\| \boldsymbol{\mathcal{X}} \times_{n\in[N]} \mathbf{Q}_n^\top \right\|_F^2, \tag{1}$$

where $\|\boldsymbol{\mathcal{X}} \times_{n\in[N]} \mathbf{Q}_n^\top\|_F^2 = \sum_{t\in[T]} \|\boldsymbol{\mathcal{X}}_t \times_{n\in[N]} \mathbf{Q}_n^\top\|_F^2$. For $d_n \leq D_n \forall n \in [N]$, (1) seeks $N$ low-rank bases to compress the tensor measurements so that the aggregate preserved variance is maximized. Tucker is commonly implemented by means of the HOSVD or HOOI algorithms. HOSVD is a single-shot method that approximates the $N$ bases in (1) disjointly by $N$ parallel PCAs of the form

$$\max_{\mathbf{Q}\in\mathbb{S}_{D_n\times d_n}} \|\texttt{mat}(\boldsymbol{\mathcal{X}}, n)^\top \mathbf{Q}\|_F^2. \tag{2}$$

On the other hand, HOOI is an iterative method which optimizes the $N$ bases jointly. In general, initialized at bases $\{\mathbf{Q}_{n,0}\}_{n\in[N]}$, at iteration $i > 0$ and for $n = 1, 2, \ldots, N$, HOOI returns $\mathbf{Q}_{n,i}$ as the solution to the PCA problem

$$\max_{\mathbf{Q}\in\mathbb{S}_{D_n\times d_n}} \|\mathbf{A}_{n,i}{}^\top \mathbf{Q}\|_F^2 \tag{3}$$

where

$$\mathbf{A}_{n,i} = \texttt{mat}(\boldsymbol{\mathcal{X}} \times_{m\in[n-1]} \mathbf{Q}_{m,i}^\top \times_{k\in[N-n]+n} \mathbf{Q}_{k,i-1}^\top, n). \tag{4}$$

### C. Outliers and L1-Tucker

Outliers appear often in datasets and can significantly compromise the performance of Tucker methods. Motivated by the success of L1-PCA in vector-data analysis [32], L1-Tucker decomposition has been proposed as an outlier-resistant Tucker reformulation. In fact, it has been documented in many data experiments that, when the processed tensor is clean, or corrupted with only benign noise, L1-Tucker attains similar performance to standard Tucker. However, when the processed tensor is corrupted with heavy-tail outliers and L2-norm based Tucker methods fail, L1-Tucker has repeatedly demonstrated marked robustness [22]–[26].

Mathematically, L1-Tucker derives by substituting the outlier-responsive L2 norm in (1) by the more robust L1-norm, as

$$\max_{\{\mathbf{Q}_n \in \mathbb{S}_{D_n \times d_n}\}_{n\in[N]}} \left\| \boldsymbol{\mathcal{X}} \times_{n\in[N]} \mathbf{Q}_n^\top \right\|_1, \tag{5}$$

where $\|\boldsymbol{\mathcal{X}} \times_{n\in[N]} \mathbf{Q}_n^\top\|_1 = \sum_{t\in[T]} \|\boldsymbol{\mathcal{X}}_t \times_{n\in[N]} \mathbf{Q}_n^\top\|_1$. L1-HOSVD [23], [33], [34] approximates the solution to L1-Tucker by $N$ parallel L1-PCA problems. That is, for every $n \in [N]$, it finds $\mathbf{Q}_n$ by solving (approximately or exactly) the L1-PCA

$$\max_{\mathbf{Q}\in\mathbb{S}_{D_n\times d_n}} \|\texttt{mat}(\boldsymbol{\mathcal{X}}, n)^\top \mathbf{Q}\|_1. \tag{6}$$

On the other hand, L1-HOOI is an iterative process that provably attains a higher L1-Tucker metric when initialized at the solution of L1-HOSVD [23], [35]. Initialized at $\{\mathbf{Q}_{n,0}\}_{n\in[N]}$ (typically by means of L1-HOSVD), at every iteration $i \geq 1$, L1-HOOI updates $\mathbf{Q}_{n,i}$ by solving

$$\max_{\mathbf{Q}\in\mathbb{S}_{D_n\times d_n}} \|\mathbf{A}_{n,i}{}^\top \mathbf{Q}\|_1, \tag{7}$$

where $\mathbf{A}_{n,i}$ is defined in (4).

Detailed complexity analyses for both L1-HOSVD and L1-HOOI are offered in [23]. The same work also presents formal convergence guarantees for the iterative L1-HOOI. As seen above, L1-HOSVD and L1-HOOI are implemented through a series of L1-PCAs. L1-PCA admits an exact solution by combinatorial optimization with high cost [32]. However, there are multiple high-performing approximate L1-PCA solvers in the literature that can be used by L1-Tucker methods. In the algorithmic developments of this work, we consider the L1-norm Bit-Flipping (L1-BF) algorithm of [36]. For the sake of completeness, a brief description of L1-BF follows.

Consider matrix $\mathbf{X} \in \mathbb{R}^{Z\times Q}$, for $Q \geq Z$, and the L1-PCA

$$\max_{\mathbf{Q}\in\mathbb{S}_{Z\times z}} \left\| \mathbf{X}^\top \mathbf{Q} \right\|_1. \tag{8}$$

L1-BF is based on the following Theorem, presented in [32].

*Theorem 1:* [32] Let $\mathbf{B}_{\text{opt}} \in \{\pm 1\}^{Q\times z}$ be a solution to $\max_{\mathbf{B}\in\{\pm1\}^{Q\times z}} \|\mathbf{X}\mathbf{B}\|_*$. Then, $\texttt{Proc}(\mathbf{X}\mathbf{B}_{\text{opt}})$ is an exact solution to L1-PCA in (8).

The nuclear norm $\|\cdot\|_*$ returns the sum of the singular values of its argument and, for any tall matrix $\mathbf{A} \in \mathbb{R}^{Z\times z}$ that admits SVD $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}_{z\times z}\mathbf{V}^\top$, $\texttt{Proc}(\mathbf{A}) = \mathbf{U}\mathbf{V}^\top$.

**Algorithm 1.** L1-Tucker Decomposition (batch-processing) [23]

**Input:** $\boldsymbol{\mathcal{X}}$, $\{\mathbf{Q}_n\}_{n\in[N]}$

Until convergence/termination

  For $n \in [N]$

    $\mathbf{A} \leftarrow \text{mat}(\boldsymbol{\mathcal{X}} \times_{m\in[n-1]} \mathbf{Q}_m^\top \times_{k\in[N-n]+n} \mathbf{Q}_k^\top, n)$

    $\mathbf{B} \leftarrow \text{sgn}(\mathbf{A}^\top \mathbf{Q}_n)$

    $\mathbf{Q}_n \leftarrow \texttt{L1-BF}(\mathbf{A}, \mathbf{B})$

**Return:** $\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_N$

**Function:** $\texttt{L1-BF}(\mathbf{A}, \mathbf{B})$, $\%\ \mathbf{B} \in \{\pm 1\}^{Q\times z}$

Until convergence/termination

  $(\bar{k}, \bar{l}) \leftarrow \underset{(k,l)\in[Q]\times[z]}{\text{argmax}} \left\|\mathbf{A}\left(\mathbf{B} - 2\mathbf{e}_{k,Q}\mathbf{e}_{l,z}^\top[\mathbf{B}]_{k,l}\right)\right\|_*$

  $\mathbf{B} \leftarrow \mathbf{B} - 2\mathbf{e}_{\bar{k},Q}\mathbf{e}_{\bar{l},z}^\top[\mathbf{B}]_{\bar{k},\bar{l}}$

$\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top \leftarrow \text{svd}(\mathbf{AB})$

$\mathbf{Q} \leftarrow \mathbf{U}\mathbf{V}^\top$

**Return:** $\mathbf{Q}$

Fig. 1. L1-Tucker Decomposition (batch-processing) [23]

In view of Theorem 1, [36] proposed to initialize at arbitrary $\mathbf{B}_0 \in \{\pm 1\}^{Q\times z}$ and iteratively conduct optimal single-bit flips (negations). Let $\mathbf{e}_{q,Q}$ denote the $q$-th column of the size-$Q$ identity matrix $\mathbf{I}_Q$. Then, at iteration $i \geq 1$, L1-BF solves

$$(k', l') = \underset{(k,l)\in[Q]\times[z]}{\text{argmax}} \left\|\mathbf{X}\left(\mathbf{B}_{i-1} - 2\mathbf{e}_{k,Q}\mathbf{e}_{l,z}^\top[\mathbf{B}_{i-1}]_{k,l}\right)\right\|_* \quad (9)$$

and updates $\mathbf{B}_i = \mathbf{B}_{i-1} - 2\mathbf{e}_{k',Q}\mathbf{e}_{l',z}[\mathbf{B}_{i-1}]_{k',l'}$. Among all possible single bit-flips, negation of the $(k', l')$-th entry of $\mathbf{B}_{i-1}$ offers the maximum possible value in $\|\mathbf{X}\mathbf{B}_i\|_*$. Importantly, L1-BF is guaranteed to monotonically increase the metric and converge in finite (in practice, few) iterations. A pseudocode of L1-Tucker, implemented by means of L1-HOOI is offered in Fig. 1.

### *D. Existing Methods for Incremental and Dynamic Tucker*

Streaming and robust matrix PCA has been thoroughly studied over the past decades [37]–[42]. However, extending matrix PCA (batch or streaming) to tensor analysis is a non-trivial task that has been attracting increasing research interest. To date, there exist multiple alternative methods for batch tensor analysis (e.g., HOSVD, HOOI, L1-HOOI) but only few for streaming/dynamic tensor analysis. For example, DTA [29], [30] efficiently approximates the HOSVD solution by processing measurements incrementally, with a fixed computational cost per update. Moreover, DTA can track multi-linear changes of subspaces, weighing past measurements with a forgetting factor. STA [29], [30] is a fast alternative to DTA, particularly designed for time-critical applications. WTA is another DTA variant which, in contrast to DTA and STA, adapts to changes by considering only a sliding window of measurements. The ALTO method was presented in [31]. For each new measurement, ALTO updates the bases through a tensor regression model. In [43], authors presented another method for Low-Rank Updates to Tucker (LRUT). When a new measurement arrives, LRUT projects it on the current bases and few more randomly chosen orthogonal directions, forming an augmented core tensor. Then it updates the bases by standard Tucker (e.g., HOSVD) on this extended core. In [44], authors consider very large tensors and propose randomized algorithms for Tucker decomposition based on the TENSORSKETCH [45]. It is stated that these algorithms can also extend for processing streaming data. Randomized methods for Tucker of streaming tensor data were also proposed in [46]. These methods rely on dimension-reduction maps for sketching the Tucker decomposition and they are accompanied by probabilistic performance guarantees. More methods for incremental tensor processing were presented in [47]–[50], focusing on specific applications such as foreground segmentation, visual tracking, and video foreground/background separation.

Methods for incremental CPD/PARAFAC tensor analysis have also been presented. For instance, authors in [51] consider the CPD/PARAFAC factorization model and assume that $N$-way measurements are streaming. They propose CP-Stream, an algorithm that efficiently updates the CPD every time a new measurement arrives. CP-stream can accommodate user-defined constraints in the factorization such as non-negativity. In addition, authors in [52] consider a Bayesian probabilistic reformulation of the CPD/PARAFAC factorization, assuming that the entries of the processed tensor are streaming across all modes, and develop a posterior inference algorithm (POST). Further, the problem of robust and incremental PARAFAC has also been studied and algorithms have been presented in [53], [54]. Typically, the application spaces of CPD and Tucker are complementary: CPD is preferred when uniqueness and interpretability are needed; Tucker allows for the latent components to be related (dense core) and it is preferred for low-rank tensor compression and completion, among other tasks [4], [14].

The problem of both outlier-resistant and dynamic Tucker analysis remains to date largely unexplored and it is the main focus of this work.

## III. Problem Statement

Focusing on outlier-resistant tensor processing, we wish to estimate the L1-Tucker bases of a tensor-data model, as formulated in (5). We assume, however, that the measurements $\{\boldsymbol{\mathcal{X}}_t\}_{t=1}^T$ are originally unavailable and collected in a streaming fashion, one at a time.

To set our algorithmic guidelines, we start by considering two simplistic antipodal approaches. On the one hand, an *instantaneous* approach would L1-Tucker-decompose each new measurement to return new bases, independently of any previously seen data. While this approach is memoryless and computationally simple, its bases estimation performance is bound to be limited, especially in low Signal-to-Noise Ratio (SNR). On the other hand, an *increasing-batch* approach would append the new measurement to the already collected ones and re-solve the L1-Tucker problem from scratch. As the data collection increases, this method could attain superior bases estimation performance at the expense of increasingly high computational and storage overhead.

Both these extreme approaches exhibit an unfavorable performance/cost trade-off. In contrast, a preferred method would leverage each new measurement, together with previous ones, to efficiently update the existing bases. The development of such a method is the main contribution of this paper, as presented in detail in the following Section IV.

## IV. Proposed Algorithm

The proposed *Dynamic L1-Tucker Decomposition (D-L1-Tucker)* is a method for incremental estimation of the L1-Tucker bases. D-L1-Tucker is designed to (i) attain high bases estimation performance, (ii) suppress outliers, and (iii) adapt to changes of the nominal subspaces. In this Section, we present D-L1-Tucker in detail, addressing bases initialization, bases updates, parameter tuning, and modifications for long-term efficiency.

### A. Batch Initialization

Considering the availability of an initial batch of $B \ll T$ measurements, $\mathcal{B} = \{\boldsymbol{\mathcal{X}}_1, \ldots, \boldsymbol{\mathcal{X}}_B\}$, we run on it L1-HOSVD or L1-HOOI to obtain an initial set of L1-Tucker estimates $\mathcal{Q}_0 = \{\mathbf{Q}_1^{(0)}, \ldots, \mathbf{Q}_N^{(0)}\}$.

Apart from $\mathcal{Q}_0$, we also initialize a *memory set* $\mathcal{M}_0 = \Omega(\mathcal{B}, M)$, for some maximum memory size $M \geq 0$. For any ordered set $\mathcal{I}$ and integer $Z \geq 0$, we define

$$\Omega(\mathcal{I}, Z) = \begin{cases} \mathcal{I}, & \text{if } \ |\mathcal{I}| \leq Z, \\ [\mathcal{I}]_{|\mathcal{I}|-Z+1:|\mathcal{I}|}, & \text{if } \ |\mathcal{I}| > Z, \end{cases} \tag{10}$$

where $|\cdot|$ denotes the cardinality (number of elements in a set) of its input argument. That is, $\Omega(\mathcal{B}, M)$ returns the last $\min\{M, B\}$ elements in $\mathcal{B}$.

If an initialization batch $\mathcal{B}$ is not available, the bases in $\mathcal{Q}_0$ are chosen arbitrarily and the initial memory $\mathcal{M}_0$ is empty. In this case, D-L1-Tucker becomes purely streaming.

### B. Streaming Updates

When a new measurement $\bar{\boldsymbol{\mathcal{X}}}_t \neq \mathbf{0}$, $t \geq 1$, is collected,[1] we first perform a check on it to assess its reliability based on the most recently updated set of bases $\mathcal{Q}_{t-1}$. Motivated by [28], [41], [55], [56], we define the reliability as

$$r_t = \left\| \bar{\boldsymbol{\mathcal{X}}}_t \times_{n\in[N]} \mathbf{Q}_n^{(t-1)^\top} \right\|_F^2 \left\| \bar{\boldsymbol{\mathcal{X}}}_t \right\|_F^{-2} \in [0, 1]. \tag{11}$$

After some algebraic manipulations, (11) can be rewritten as

$$r_t = \cos^2(\phi(\mathrm{vec}(\bar{\boldsymbol{\mathcal{X}}}_t \times_{n\in[N]} \mathbf{Q}_n^{(t-1)} \mathbf{Q}_n^{(t-1)^\top}), \mathrm{vec}(\bar{\boldsymbol{\mathcal{X}}}_t))), \tag{12}$$

where $\phi(\cdot, \cdot)$ returns the angle between its two vector arguments. Intuitively, $r_t$ quantifies how much measurement $\bar{\boldsymbol{\mathcal{X}}}_t$ conforms to the multi-way subspace spanned by $\{\mathbf{Q}_n^{(t-1)}\}_{n\in[N]}$, or, the angular proximity of $\mathrm{vec}(\bar{\boldsymbol{\mathcal{X}}}_t \times_{n\in[N]} \mathbf{Q}_n^{(t-1)} \mathbf{Q}_n^{(t-1)^\top})$ to $\mathrm{vec}(\bar{\boldsymbol{\mathcal{X}}}_t)$. This check of reliability/conformity inherits its robustness from the L1-Tucker-derived bases upon which it is defined. Moreover, if an outlier happens to pass the reliability check, L1-Tucker will try to suppress it, providing again robust bases. By definition, the value of $r_t$ will be between 0 and 1. If $r_t = 1$, then the bases in $\mathcal{Q}_{t-1}$ perfectly describe $\bar{\boldsymbol{\mathcal{X}}}_t$. In contrast, if $r_t = 0$, then the set $\mathcal{Q}_{t-1}$ does not capture any component of $\bar{\boldsymbol{\mathcal{X}}}_t$. Then, we introduce a user-defined parameter $\tau$ and consider that $\bar{\boldsymbol{\mathcal{X}}}_t$ is reliable for processing if $r_t \geq \tau$. Otherwise, $\bar{\boldsymbol{\mathcal{X}}}_t$ is considered to be an outlier and it is rejected.

[1] A bar over a tensor denotes that it is streaming.

If $\bar{\boldsymbol{\mathcal{X}}}_t$ passes the reliability check, we use it to update the bases and memory as follows. First, we append the new measurement to the most recent memory set $\mathcal{M}_{t-1}$ by computing the extended memory $\mathcal{M}' = \Phi(\mathcal{M}_{t-1}, \bar{\boldsymbol{\mathcal{X}}}_t) = \mathcal{M}_{t-1} \cup \bar{\boldsymbol{\mathcal{X}}}_t$. Then, we update the set of bases to $\mathcal{Q}_t$ by running L1-HOOI on $\mathcal{M}'$, initialized to the bases in $\mathcal{Q}_{t-1}$. Finally, we update the memory by discarding the oldest measurement, as

$$\mathcal{M}_t = \Omega(\mathcal{M}', M). \tag{13}$$

In view of the above, the cost of the L1-HOOI algorithm remains low across updates because, at any given instance, the extended memory $\mathcal{M}'$ will comprise at most $M + 1$ measurements.

If $\bar{\boldsymbol{\mathcal{X}}}_t$ fails the reliability check, we discard it and update the bases and memory by setting $\mathcal{Q}_t = \mathcal{Q}_{t-1}$ and $\mathcal{M}_t = \mathcal{M}_{t-1}$, respectively. A schematic representation of the proposed algorithm is offered in Fig. 2. Here, it is worth noting that the proposed approach focuses on temporal coherence of streaming measurements. That is, temporally sporadic points from a second nominal source of measurements could be perceived as outliers.

### C. Zero Centering

In some applications –most notable in image processing– we are interested in subspaces of zero-centered data. To this end, we can modify the proposed algorithm so that, at every update instance $(t-1)$, it computes and maintains the mean $\boldsymbol{\mathcal{C}}_{t-1} = (1/M) \sum_{m=1}^{M} [\mathcal{M}_{t-1}]_m$. Then, when $\bar{\boldsymbol{\mathcal{X}}}_t$ is collected, it will first be zero-centered as $\bar{\boldsymbol{\mathcal{X}}}_t^c = \bar{\boldsymbol{\mathcal{X}}}_t - \boldsymbol{\mathcal{C}}_{t-1}$. If $\bar{\boldsymbol{\mathcal{X}}}_t^c$ passes the reliability check, then it will be used to update the bases, as described above.

### D. Adaptation to Subspace Changes

In many applications of interest, the underlying data subspaces change across time. In such cases, an ambiguity naturally rises on whether a rejected measurement was actually an outlier or the nominal data subspaces have changed and need to be tracked. To resolve this ambiguity and allow D-L1-Tucker to adapt, we work as follows.

First, we make the minor assumption that outlying measurements appear sporadically. Then, we introduce a buffer of ambiguous measurements, $\mathcal{W}$, with capacity $W > 0$. When a streaming measurement fails the reliability check, we insert it to $\mathcal{W}$. If a measurement passes the reliability check, then we empty $\mathcal{W}$. If at any update instance $|\mathcal{W}|$ reaches $W$ –i.e., $W$ consecutive streaming measurements were rejected as outliers– then we detect a change of the nominal subspaces.

In order to adapt to these changes, we empty the memory, set $\mathcal{B} = \mathcal{W}$, and re-initialize (reset) the bases and memory, as described in Section IV-A. Next, the updates proceed as described in Sections IV-B and IV-D. A pseudocode of the proposed D-L1-Tucker algorithm is presented in Fig. 3.

### E. Long-Run Efficiency

As measurements are streaming, D-L1-Tucker keeps refining the bases estimates. Naturally, after a sufficiently

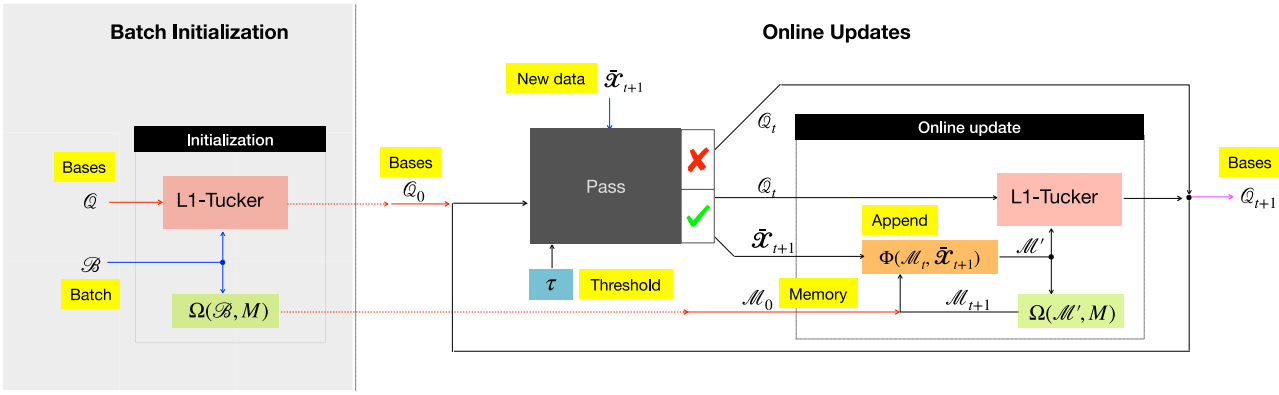



Fig. 2. A schematic illustration of the proposed algorithm for streaming L1-norm Tucker decomposition.

**Algorithm 2.** Proposed Dynamic L1-Tucker Decomposition

**Input:** $\{\bar{\boldsymbol{\mathcal{X}}}_t\}_{t\in[T]}$, $B$, $M$, $W$, $\tau$, $\mathcal{Q} \leftarrow \cup_{n\in[N]}\mathbf{Q}_n$

$\mathcal{B} \leftarrow \bigcup_{t\in[B]} \bar{\boldsymbol{\mathcal{X}}}_t$

$(\mathcal{Q}, \mathcal{M}, \mathcal{W}) \leftarrow$ `batch-init`$(\mathcal{B}, \mathcal{Q}, M)$

For $t > B$

  $(\mathcal{Q}, \mathcal{M}, \mathcal{W}) \leftarrow$ `online-updates`$(\bar{\boldsymbol{\mathcal{X}}}_t, \mathcal{M}, \mathcal{Q}, \mathcal{W}, \tau)$

**Return:** $\mathcal{Q} \rightarrow \{\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_N\}$

**Function:** `batch-init`$(\mathcal{B}, \mathcal{Q}, M)$

$\mathcal{Q} \leftarrow$ `L1-Tucker`$(\mathcal{B}, \mathcal{Q})$

$\mathcal{M} \leftarrow \Omega(\mathcal{B}, M)$

$\mathcal{W} \leftarrow \varnothing$

**Return:** $\mathcal{Q}, \mathcal{M}, \mathcal{W}$

**Function:** `online-updates`$(\bar{\boldsymbol{\mathcal{X}}}_t, \mathcal{M}, \mathcal{Q}, \mathcal{W}, \tau)$

$r_t \leftarrow \|\bar{\boldsymbol{\mathcal{X}}}_t \times_{n\in[N]} \mathbf{Q}_n^\top\|_F^2 \|\bar{\boldsymbol{\mathcal{X}}}_t\|_F^{-2}$

If $r_t > \tau$

  $\mathcal{M}' \leftarrow \Phi(\mathcal{M}, \bar{\boldsymbol{\mathcal{X}}}_t)$

  $\mathcal{Q} \leftarrow$ `L1-Tucker`$(\mathcal{M}', \mathcal{Q})$

  $\mathcal{M} \leftarrow \Omega(\mathcal{M}', M)$

  $\mathcal{W} \leftarrow \varnothing$

Else

  $\mathcal{W} \leftarrow \mathcal{W} \cup \bar{\boldsymbol{\mathcal{X}}}_t$

  If $|\mathcal{W}| = W$

    $(\mathcal{Q}, \mathcal{M}, \mathcal{W}) \leftarrow$ `batch-init`$(\mathcal{W}, \mathcal{Q}, M)$

**Return:** $\mathcal{Q}, \mathcal{M}, \mathcal{W}$

Fig. 3. Proposed Dynamic L1-Tucker Decomposition

large number of measurements have been processed, the enhancement rate of the bases estimates can be so low that does not justify the computational effort expended for the update.

In view of this observation, we can enhance the long-run efficiency of D-L1-Tucker by introducing an exponentially decreasing probability $\rho_t$ to determine whether or not the $t$-th measurement will be processed. Intuitively, when a large number of reliable measurements have been processed, $\rho_t$ should be low enough to limit the number of updates performed. For example, let us denote by $\alpha_{t-1}$ the number of consecutive measurements that have passed the reliability check at update instance $t-1$. Then, if $\bar{\boldsymbol{\mathcal{X}}}_t$ passes the reliability check, it will be processed with probability $\rho_t = \rho^{\alpha_{t-1}+1}$, for some initial probability $\rho > 0$, close to 1. If $\bar{\boldsymbol{\mathcal{X}}}_t$ fails the reliability check, then it is rejected and $\alpha_t$ is reset to 0.

### F. Parameter Configuration

The performance of D-L1-Tucker largely depends on three parameters: the initialization batch size $B$, the memory size $M$, and the reliability threshold $\tau$. Here, we discuss how to select these parameters.

*Batch size $B$:* $B$ determines the quality of the initial set of bases. That is, higher values of $B$ will generally offer better set of bases. Naturally, a very large $B$ would contradict the streaming nature of the method.

*Memory size $M$:* $M$ determines how many measurements L1-Tucker will process at each time instance. Similar to $B$, higher values of $M$ can enable superior estimation performance. At the same time, high values of $M$ increase the overhead of storage and computation (cost of L1-Tucker updates). Thus, a rule of thumb is to set $M$ as high as the storage/computation limitations of the application permit.

*Reliability threshold $\tau$:* For $\tau = 0$, all measurements will be processed (including outliers); for $\tau = 1$, all measurements will fail the reliability check and no bases updates will take place. Appropriate tuning of $\tau$ between 0 and 1 may ask for some prior knowledge on the SNR quality of the nominal data. Alternatively, in the sequel we present a data-driven method for setting $\tau$.

We start with the reasonable assumption that the initialization batch $\mathcal{B}$ is outlier-free. Then, we conduct on $\mathcal{B}$ a leave-one-out cross-validation to tune $\tau$. For every $i \in [B]$, we first form $\mathcal{B}_i = \mathcal{B}\backslash\boldsymbol{\mathcal{X}}_i$. Then, we obtain the set of bases $\mathcal{Q}_i$ by running L1-HOOI on $\mathcal{B}_i$. Next, we capture in $r_i$ the reliability of $\boldsymbol{\mathcal{X}}_i$ evaluated on $\mathcal{Q}_i$ (notice that $\boldsymbol{\mathcal{X}}_i$ did not participate in the computation of $\mathcal{Q}_i$). Finally, we set $\tau$ to the minimum, median, or maximum value of the cross-validated reliabilities $\{r_1, \ldots, r_B\}$, depending on the noise-tolerance/outlier-robustness level that we want to enforce.

## V. Experimental Studies

### A. Testing Parameter Configurations

We first study the performance of the proposed D-L1-Tucker algorithm across varying parameter configurations. We consider $T$ $N$-way measurements $\bar{\boldsymbol{\mathcal{X}}}_1, \ldots, \bar{\boldsymbol{\mathcal{X}}}_T$, where

$$\bar{\boldsymbol{\mathcal{X}}}_t = \boldsymbol{\mathcal{G}}_t \times_{n\in[N]} \mathbf{Q}_n^{\text{nom}} + \boldsymbol{\mathcal{N}}_t + \boldsymbol{\mathcal{O}}_t \in \mathbb{R}^{D\times D\times\ldots\times D}, \quad (14)$$

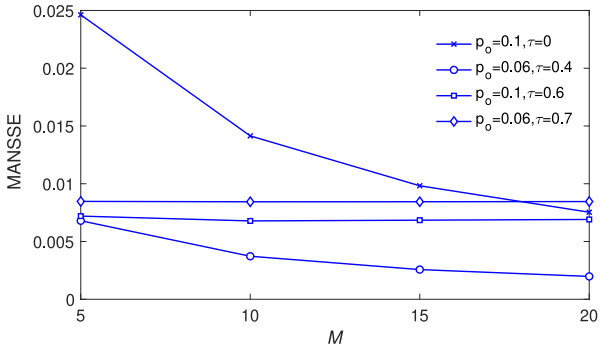



Fig. 4. $N = 3, D = 10, d = 5, B = 5, T = 30$, SNR $= 0$dB, ONR $= 14$dB, 3000 realizations.

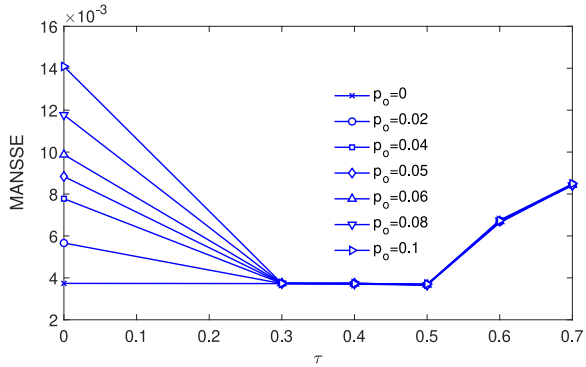



Fig. 5. $N = 3$, $D = 10$, $d = 5$, $B = 5$, $M = 10$, $T = 30$, SNR $= 0$dB, ONR $= 14$dB, 3000 realizations.

for a nominal set of bases $\mathcal{Q}^{\text{nom}} = \{\mathbf{Q}_n^{\text{nom}} \in \mathbb{S}_{D\times d}\}_{n\in[N]}$. The core tensor $\boldsymbol{\mathcal{G}}_t \in \mathbb{R}^{d\times d\times\ldots\times d}$ draws entries independently from $\mathcal{N}(0, \sigma_s^2)$. $\boldsymbol{\mathcal{N}}_t$ models Additive White Gaussian Noise (AWGN) and draws entries from $\mathcal{N}(0, \sigma_n^2)$. $\boldsymbol{\mathcal{O}}_t$ models sporadic heavy outlier corruption and is non-zero with probability $p_o$. When non-zero, $\boldsymbol{\mathcal{O}}_t$ draws entries from $\mathcal{N}(0, \sigma_o^2)$. In order to measure data quality, we define the SNR as

$$\text{SNR} = \frac{\mathbb{E}\{\|\boldsymbol{\mathcal{X}}_t\|_F^2\}}{\mathbb{E}\{\|\boldsymbol{\mathcal{N}}_t\|_F^2\}} = \frac{\sigma_s^2}{\sigma_n^2}\left(\frac{d}{D}\right)^N \tag{15}$$

and the Outlier-to-Noise Ratio (ONR)

$$\text{ONR} = \frac{\mathbb{E}\{\|\boldsymbol{\mathcal{O}}_t\|_F^2\}}{\mathbb{E}\{\|\boldsymbol{\mathcal{N}}_t\|_F^2\}} = \frac{\sigma_o^2}{\sigma_n^2}. \tag{16}$$

Our objective is to recover $\mathcal{Q}^{\text{nom}}$ by processing the measurements $\{\bar{\boldsymbol{\mathcal{X}}}_t\}_{t\in[T]}$ in a streaming way. Denoting by $\hat{\mathbf{Q}}_n$ the estimate of $\mathbf{Q}_n^{\text{nom}}$, we quantify performance by means of the Mean Aggregate Normalized Subspace Squared Error (MANSSE)

$$\text{MANSSE} = \frac{1}{2Nd}\sum_{n\in[N]}\left\|\mathbf{Q}_n^{\text{nom}}\mathbf{Q}_n^{\text{nom}\top} - \hat{\mathbf{Q}}_n\hat{\mathbf{Q}}_n^\top\right\|_F^2. \tag{17}$$

First, we set $N = 3$, $D = 10$, $d = 5$, $B = 5$, and $T = 30$. Moreover, we set $\sigma_s^2$, $\sigma_n^2$, and $\sigma_o^2$ such that SNR $= 0$dB and ONR $= 14$dB. In Fig. 4, we plot the MANSSE metric versus varying $M \in \{5, 10, 15, 20\}$ and fixed $(p_0, \tau) \in \{(0.1, 0), (0.06, 0.4), (0.1, 0.6), (0.06, 0.7)\}$. We observe that the curves corresponding to $\tau \geq 0.6$ are almost horizontal. This implies that these values of $\tau$ are too strict, rejecting almost all measurements. For $\tau = 0$, all measurements are processed (outliers and nominal ones); therefore, we see that the estimation performance improves as $M$ increases, however, the estimation error is somewhat high because of the processed outliers. The curve corresponding to $\tau = 0.4$ exhibits the best performance across the board.

Next, motivated by the above observations, we fix SNR $= 0$dB and let $\tau$ vary in $\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$. In Fig. 5, we plot the MANSSE versus $\tau$ for different values of outlier probability $p_o$. We notice that for any $\tau \in [0.3, 0.5]$, D-L1-Tucker exhibits high, almost identical MANSSE performance independently of $p_o$. This, in turn, suggests that the SNR plays an important role in determining the optimal value of $\tau$, for which nominal measurements will be processed and outliers will be rejected with high probability. For the same study, we present the frequency of rejection versus $\tau$ in Fig. 6. Again, we notice that for very low values of $\tau$, most measurements are accepted for processing. In contrast, for very high values of $\tau$, most measurements are rejected. Interestingly, this figure suggests that for any given parameter configuration there will be an optimal value of $\tau$ for which the frequency of rejection will approach the probability of outliers $p_o$ –which, in turn, implies that in general outliers will rejected and nominal data will be processed.

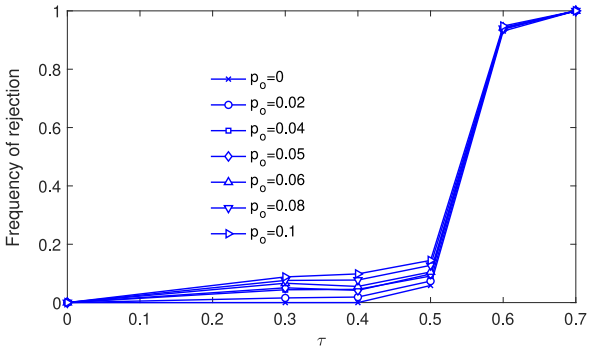



Fig. 6. $N = 3$, $D = 10$, $d = 5$, $B = 5$, $M = 10$, $T = 30$, SNR $= 0$dB, ONR $= 14$dB, 3000 realizations.

Finally, we let $p_o$ vary in $\{0, 0.02, 0.06, 0.08, 0.1\}$ and, in Fig. 7, we plot the frequency of rejection versus $p_o$. In accordance with previous observations, we see that high values of $\tau$ result in high rejection frequency, independently of the value of $p_o$. Interestingly, we see that values of $\tau$ within $[0.3, 0.4]$ appear to be near-optimal for this particular SNR and parameter configuration, as their performance almost coincides with the $45^\circ$ slope, at all points of which the frequency of rejection equals the outlier probability $p_o$.

### B. Convergence

At any fixed update index, D-L1-Tucker is guaranteed to converge. That is, when a measurement is deemed reliable,

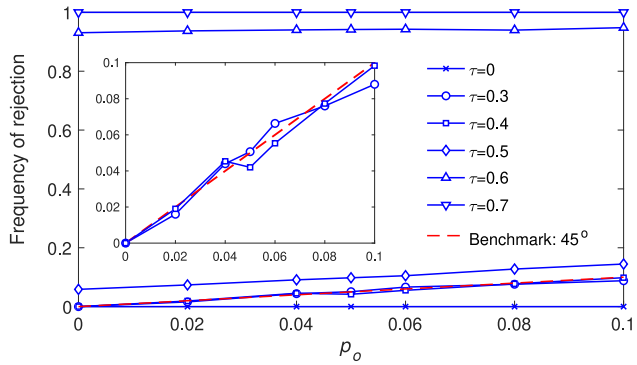



Fig. 7. $N = 3$, $D = 10$, $d = 5$, $B = 5$, $M = 10$, $T = 30$, SNR $= 0$dB, ONR $= 14$dB, 3000 realizations.

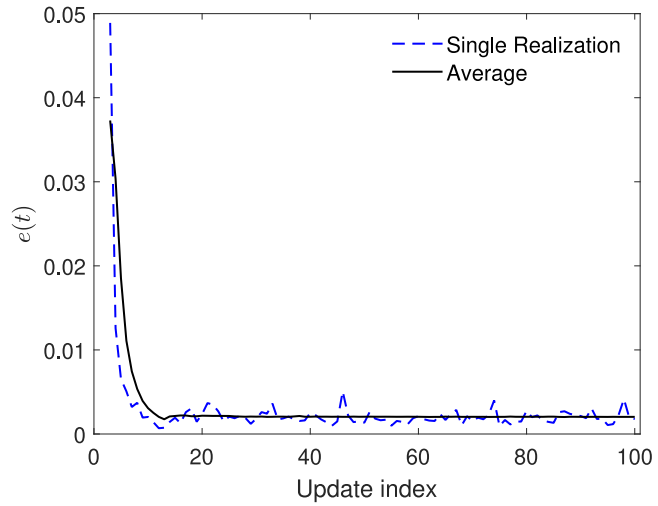



Fig. 8. Empirical convergence. $N = 3$, $D = 10$, $d = 3$, $T = 100$, $B = 2$, $M = 12$, $W = 0$, SNR $= -6$dB, data-driven $\tau$, 20 000 realizations.

the proposed algorithm processes the measurements in memory after appending the new measurement by means of L1-HOOI the convergence of which has been formally proven [23]. Further, due to the sturdiness of L1-Tucker it is expected that, after many updates, the replacement of a single nominal measurement in the memory set will not cause much of a shift to the bases. To illustrate this, we conduct the following study. We process measurements $\bar{\boldsymbol{\mathcal{X}}}_1, \ldots, \bar{\boldsymbol{\mathcal{X}}}_T$ in the form of (14). D-L1-Tucker returns $\hat{\mathcal{Q}}_t = \{\hat{\mathbf{Q}}_{n,t} \in \mathbb{S}_{D\times d}\}_{n\in[N]}$. In order to evaluate convergence across updates, we measure $e(t) = \frac{1}{2Nd}\sum_{n\in[N]} \|\hat{\mathbf{Q}}_{n,t}\hat{\mathbf{Q}}_{n,t}^\top - \hat{\mathbf{Q}}_{n,t-1}\hat{\mathbf{Q}}_{n,t-1}^\top\|_F^2$. In Fig. 8, we plot $e(t)$ versus update index $t$ for a single realization of measurements. Moreover, we plot $e(t)$ when it is sample-average computed over 20 000 statistically independent realizations of measurements. As expected, we see that after enough measurements have been processed, $e(t)$ remains low and very close to the average expected performance. We conclude that, upon nominal operation and large enough $M$, bases changes will be minuscule in the long run. This will be even more emphatic for high SNR. Finally, the long-run efficiency feature of D-L1-Tucker, introduced in Section IV-E, can also enforce convergence/termination.

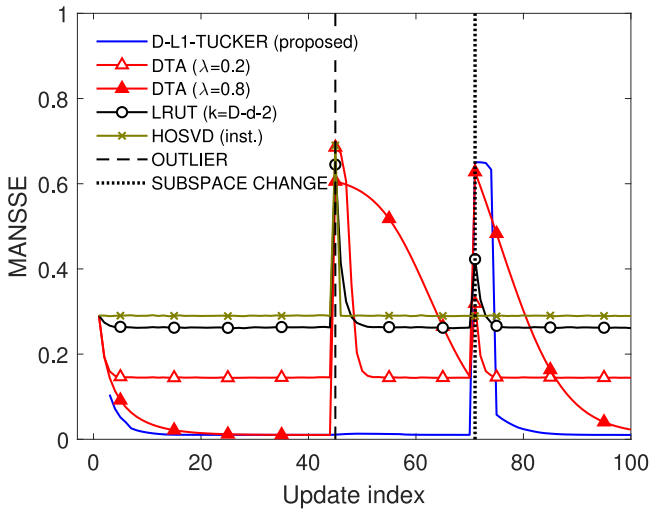



Fig. 9. $N = 3$, $D = 10$, $d = 3$, $T_1 = 70$, $T_2 = 30$, $B = 2$, $M = 12$, $W = 4$, SNR $= -6$dB, ONR $= 18$dB, data-driven $\tau$, 20 000 realizations.

### C. Dynamic Subspace Adaptation

We consider a total of $T = T_1 + T_2$ streaming measurements, in the form of (14). The first $T_1$ measurements are generated by nominal bases $\mathcal{Q}^{\text{nom},1}$. For $t > T_1$ and on, the measurements are generated by bases $\mathcal{Q}^{\text{nom},2}$. The angular proximity of $\mathbf{Q}_n^{\text{nom},1}$ to $\mathbf{Q}_n^{\text{nom},2}$, defined as $1 - \|\mathbf{Q}_n^{\text{nom},1}\mathbf{Q}_n^{\text{nom},1\top} - \mathbf{Q}_n^{\text{nom},2}\mathbf{Q}_n^{\text{nom},2\top}\|_F^2(2d)^{-1}$, is set between 30% and 40% for every $n \in [N]$. Moreover, we consider that the outlier is only active at instance $t = t_o = 45$. We set $N = 3$, $D = 10$, $d = 3$, $T_1 = 70$, and $T_2 = 30$. The SNR and ONR are set to $-6$dB and 18 dB, respectively. We process all measurements by the proposed D-L1-Tucker algorithm for $B = 2$, $M = 12$, $W = 4$, and data-driven $\tau$ (median of cross-validated batch reliabilities). We also process the streaming measurements with DTA ($\lambda = 0.2, 0.8$), LRUT (additional core dimensions $k = D - d - 2$), and instantaneous HOSVD[2] counterparts.

In Fig. 9, we plot the MANSSE versus update index $t$. All methods, except for the instantaneous HOSVD, start from a higher MANSSE value and refine their bases by processing streaming measurements until they reach a low plateau. At $t = 45$, when the outlier appears, we observe that all competing methods suffer a significant performance loss. In contrast, the proposed D-L1-Tucker algorithm discards the outlier and its performance remains unaffected. When subsequent measurements are streaming, the competing methods start recovering until they reach again a low plateau, which is largely determined by the SNR and the parameter configuration of each method. Interestingly, the instantaneous HOSVD recovers rapidly, after just one measurement, because it is memoryless. DTA ($\lambda = 0.2$) recovers faster than DTA ($\lambda = 0.8$) but its MANSSE plateau is

[2]At update instance $t$, instantaneous HOSVD returns the HOSVD solution of $\bar{\boldsymbol{\mathcal{X}}}_t$, independently of any previous measurements.

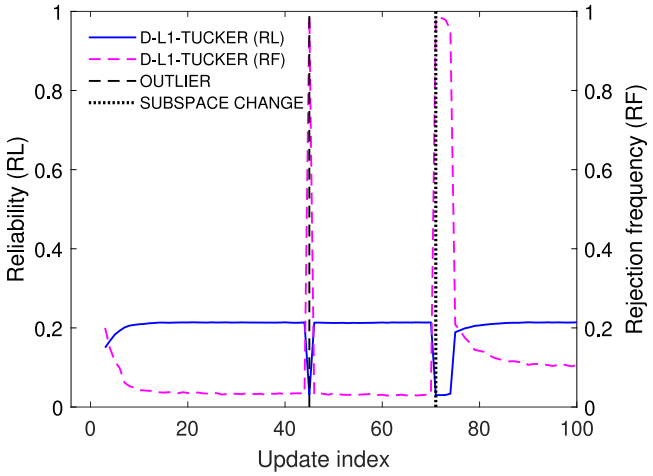



Fig. 10. $N = 3, D = 10, d = 3, T_1 = 70, T_2 = 30, B = 2, M = 12, W = 4$, SNR $= -6$dB, ONR $= 18$dB, data-driven $\tau$, 20 000 realizations.

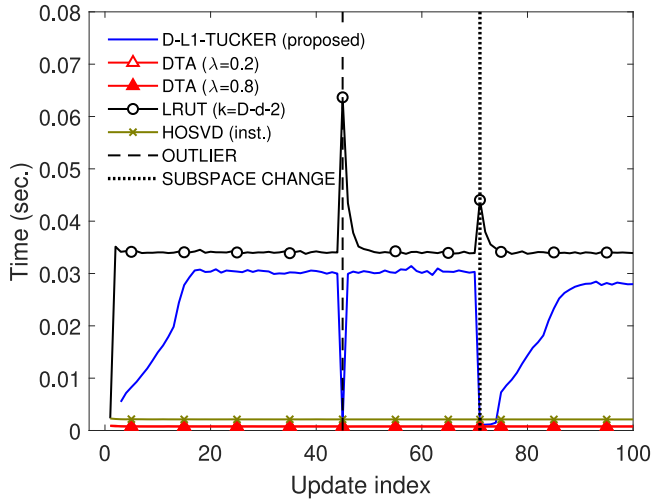



Fig. 11. $N = 3, D = 10, d = 3, T_1 = 70, T_2 = 30, B = 2, M = 12, W = 4$, SNR $= -6$dB, ONR $= 18$dB, data-driven $\tau$, 20 000 realizations.

higher. LRUT also recovers and reaches its plateau performance after it has seen about 10 measurements after the outlier. At time instance 71 the nominal data subspaces shift, affecting all methods expect for the memoryless/instantaneous HOSVD. D-L1-Tucker attains a high value of MANSSE for about $W$ time instances while its ambiguity window is being filled. Right after, it rapidly recovers to a low MANSSE value and keeps refining as more measurements are streaming. DTA and LRUT are also adapting to the new underlying structure after processing a few measurements. Another interesting observation is that the low plateau level for each method appears to be the same in the two distinct coherence windows.

In Fig. 10, we plot the reliability of the streaming measurements across updates in accordance with (11). At the same figure, we illustrate the frequency of rejection; that is, the frequency by which measurements fail the reliability check. We notice that the outlier at $t = 45$ and the $W$ measurements following the change of subspaces are rejected with probability close to 1. In addition, we observe the instantaneous reliability drop when the outlier appears and when nominal subspaces change. For this value of SNR $= -6$dB, the reliability level for nominal measurements is about 0.2 and our data-driven $\tau$ is accordingly low.

We conclude this study by comparing the run time of each method across updates. In Fig. 11, we plot the instantaneous run times. We observe that the instantaneous HOSVD and DTA exhibit constant run time across updates independently of outliers or changes of subspaces. D-L1-Tucker also exhibits about constant runtime after its memory has been filled. Moreover, we notice an instantaneous drop in the runtime at index $t = 45$ which is because D-L1-Tucker discarded the outlier and did not process it. In contrast, when the outlier appears and when the subspaces change, LRUT attains an increase in runtime, as it tries to adapt.

Next, we repeat the above study. This time, instead of having a fixed outlier at an index, every measurement with index $t > B$ is outlier corrupted with probability $p_o = 0.1$. Moreover, $T_1 = 75$ and $T_2 = 85$. This time, we include a curve which corresponds

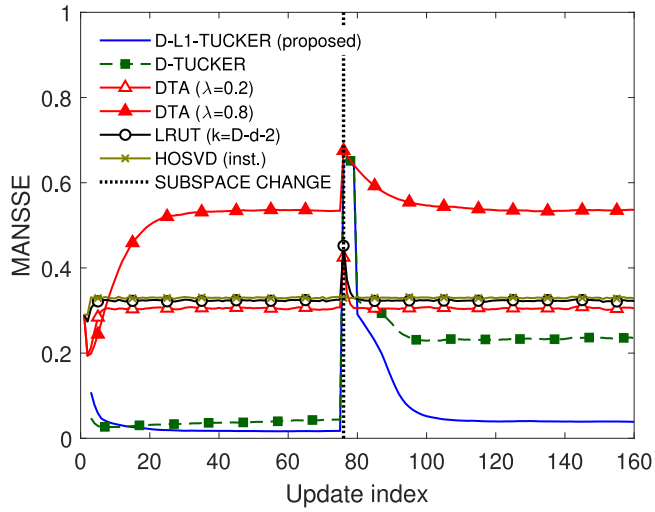



Fig. 12. $N = 3, D = 10, d = 3, T_1 = 75, T_2 = 85, B = 2, M = 12, W = 4$, SNR $= -6$dB, ONR $= 18$dB, data-driven $\tau$, 20 000 realizations.

to a method we label D-Tucker. D-Tucker is identical to D-L1-Tucker with the exception that standard Tucker by means of HOOI is employed instead of L1-Tucker.

In Fig. 12, we plot the MANSSE versus update index. We observe that the estimation performance of the DTA curves degrades due to the outliers until a plateau is reached. Their estimation error increases at the subspaces change index and returns to its plateau performance after a few measurements. Expectedly, the instantaneous HOSVD appears to exhibit constant performance for any index $t > B$. A similar observation is made for the LRUT curve with the exception of update indices 75 to 80 where it adjusts to the new underlying data structure. D-Tucker starts from a low MANSSE value and improves for a while, however, its performance slowly drops as measurements are streaming. This is because outliers are passing the reliability check of the L2-norm derived bases which, in turn, affects the

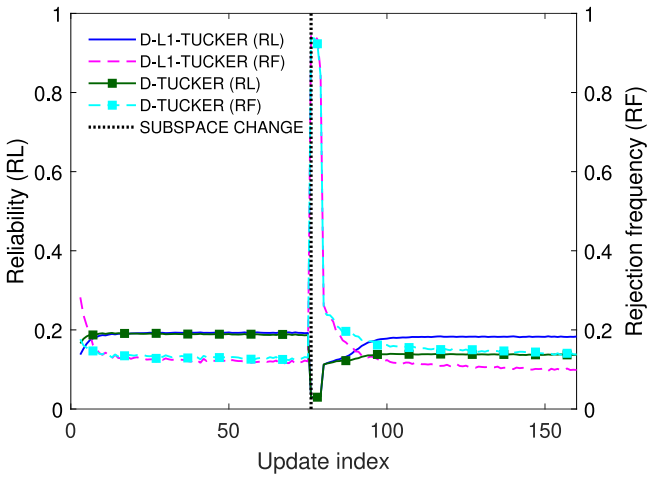



Fig. 13. $N = 3, D = 10, d = 3, T_1 = 75, T_2 = 85, B = 2, M = 12, W = 4$, SNR $= -6$dB, ONR $= 18$dB, data-driven $\tau$, 20 000 realizations.

performance of the memory batch processing. In contrast, we see that D-L1-Tucker keeps improving its performance up to update index 75 where the underlying data structure changes. Then, after the ambiguity batch windows of D-Tucker and D-L1-Tucker are filled, they both reset based on the measurements in their window –each measurement of which is outlier corrupted with probability $p_o$. Due to its inherent L1-norm robustness, D-L1-Tucker is able to recover and learn the underlying data structure of the new coherence window. In contrast, we see that D-Tucker fails to recover due to the outlying measurements in its ambiguity window.

In Fig. 13, we illustrate the reliability values and rejection frequencies as they were computed by the proposed D-L1-Tucker and its counterpart D-Tucker. We observe that during the first coherence window ($t \leq T_1$), both methods exhibit almost identical reliabilities and rejection frequencies. However, in view of Fig. 12, we infer that D-L1-Tucker is more successful at rejecting outliers compared to D-Tucker. A few measurements after the change of subspaces at index $t = 75$, we see that D-L1-Tucker converges to reliability and rejection frequency values similar to those of the first coherence window which implies that it has adapted nicely to the new coherence window. On the other hand, the reliability and rejection frequency values to which D-Tucker converges in the second coherence window, both diverge from their corresponding values in the first coherence window which, in turn, implies that D-Tucker was not able to adapt well. This is also depicted in Fig. 12 where we see that in the second coherence window D-Tucker converged to a high MANSSE value.

### *D. Dynamic Video Foreground/Background Separation*

Foreground-background separation is a common task in video processing applications, including moving object tracking, security surveillance, and traffic monitoring. The omnipresent background in a static camera scene determines a nominal subspace, while any foreground movement –e.g., by vehicles or people– represent intermittent outliers. In this experiment, we use D-L1-Tucker to estimate the background and foreground of incoming video frames and compare its performance with that of state-of-the-art alternatives.

For this experiment, we use videos from the CAVIAR database [57]. Specifically, we use two videos, each containing 100 frames of size $(D_1 = 173) \times (D_2 = 231)$ and capturing a different scene. Each video is viewed as collection of frames – i.e., 2-way tensor measurements. We collate the two videos, one behind the other to form the data stream $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{D_1 \times D_2 \times (T=200)}$. Below, we denote by $\bar{\boldsymbol{\mathcal{X}}}_t$ the $t$-th frontal slab of $\boldsymbol{\mathcal{X}}$ (i.e., the $t$-th video frame).

We apply the proposed algorithm on $\boldsymbol{\mathcal{X}}$ by setting $d_n = d = 3 \forall n \in [2]$, $B = 5$, $M = 10$, $W = 20$, and $\tau$ by the proposed batch reliability cross-validation. For every $t \in [T - B] + B$, we obtain bases $\mathbf{Q}_1^{(t)}$ and $\mathbf{Q}_2^{(t)}$ and the mean frame $\boldsymbol{\mathcal{C}}_t$. Accordingly, we estimate the background as $\boldsymbol{\mathcal{X}}_t^{\text{BG}} = \mathbf{Q}_1^{(t)} \mathbf{Q}_2^{(t)\top} (\bar{\boldsymbol{\mathcal{X}}}_t - \boldsymbol{\mathcal{C}}_t) \mathbf{Q}_2^{(t)} \mathbf{Q}_2^{(t)\top} + \boldsymbol{\mathcal{C}}_t$ and the foreground as $\boldsymbol{\mathcal{X}}_t^{\text{FG}} = \boldsymbol{\mathcal{X}}_t - \boldsymbol{\mathcal{X}}_t^{\text{BG}}$.

We compare the performance of the proposed algorithm with that of DTA, LRUT, OSTD,[3] HOOI (increasing batch), and L1-HOOI (increasing batch). For the last two benchmark approaches, at any frame index $t$, we run HOOI/L1-HOOI on the entire batch $\{\bar{\boldsymbol{\mathcal{X}}}_j\}_{j \in [t]}$, starting from arbitrary initialization. We notice that DTA is capable of tracking scene changes using a forgetting factor $\lambda$. Since the background estimation involves mean subtraction, for a fair comparison with the proposed method, we enable mean tracking for DTA by computing $\boldsymbol{\mathcal{C}}_t^{\text{DTA}} = \lambda \boldsymbol{\mathcal{C}}_{t-1}^{\text{DTA}} + (1 - \lambda) \bar{\boldsymbol{\mathcal{X}}}_t$, for $\boldsymbol{\mathcal{C}}_1^{\text{DTA}} = \bar{\boldsymbol{\mathcal{X}}}_1$. For all other methods, we compute the mean incrementally at any $t$ as $\boldsymbol{\mathcal{C}}_t = ((t-1)\boldsymbol{\mathcal{C}}_{t-1} + \bar{\boldsymbol{\mathcal{X}}}_t)/t$. For DTA, we use two values of forgetting factor, $\lambda = 0.95, 0.7$ and for LRUT we set the number of additional core dimensions to $k_n = D_n - d - 3 \; \forall n \in [2]$.

In Fig. 14 and Fig. 15, we present the backgrounds and foregrounds obtained by the proposed method and existing counterparts at the 75-th frame (scene 1) and the 150-th frame (scene 2), respectively. We observe from Fig. 14 that HOOI (increasing batch), LRUT, and OSTD perform similarly leaving a trail of a ghostly appearance behind the person in their respective foreground frames. We notice that OSTD and L1-HOOI (increasing batch) perform better with a smoother trail behind the person in their foreground frames. DTA with $\lambda = 0.7$ captures the person in its background, leading to an undesirably smudged foreground estimate. DTA with $\lambda = 0.95$ demonstrates a cleaner foreground estimate, similar to that of the adaptive mean (background estimated by the same adaptive mean that we use for DTA), however their backgrounds contain a ghostly appearance of the person. The proposed method extracts a cleaner background and foreground owing to its outlier rejection capability.

In Fig. 15, we demonstrate the performance after the scene changes at $t = 100$ by presenting the estimated backgrounds and foregrounds at frame index $t = 150$. We observe that HOOI, L1-HOOI, OSTD, and LRUT perform poorly because they are not designed to track changes in the scene. DTA with $\lambda = 0.95$

[3]OSTD –i.e., Online Stochastic Tensor Decomposition– was specifically designed for background/foreground separation in multispectral video sequences [50].

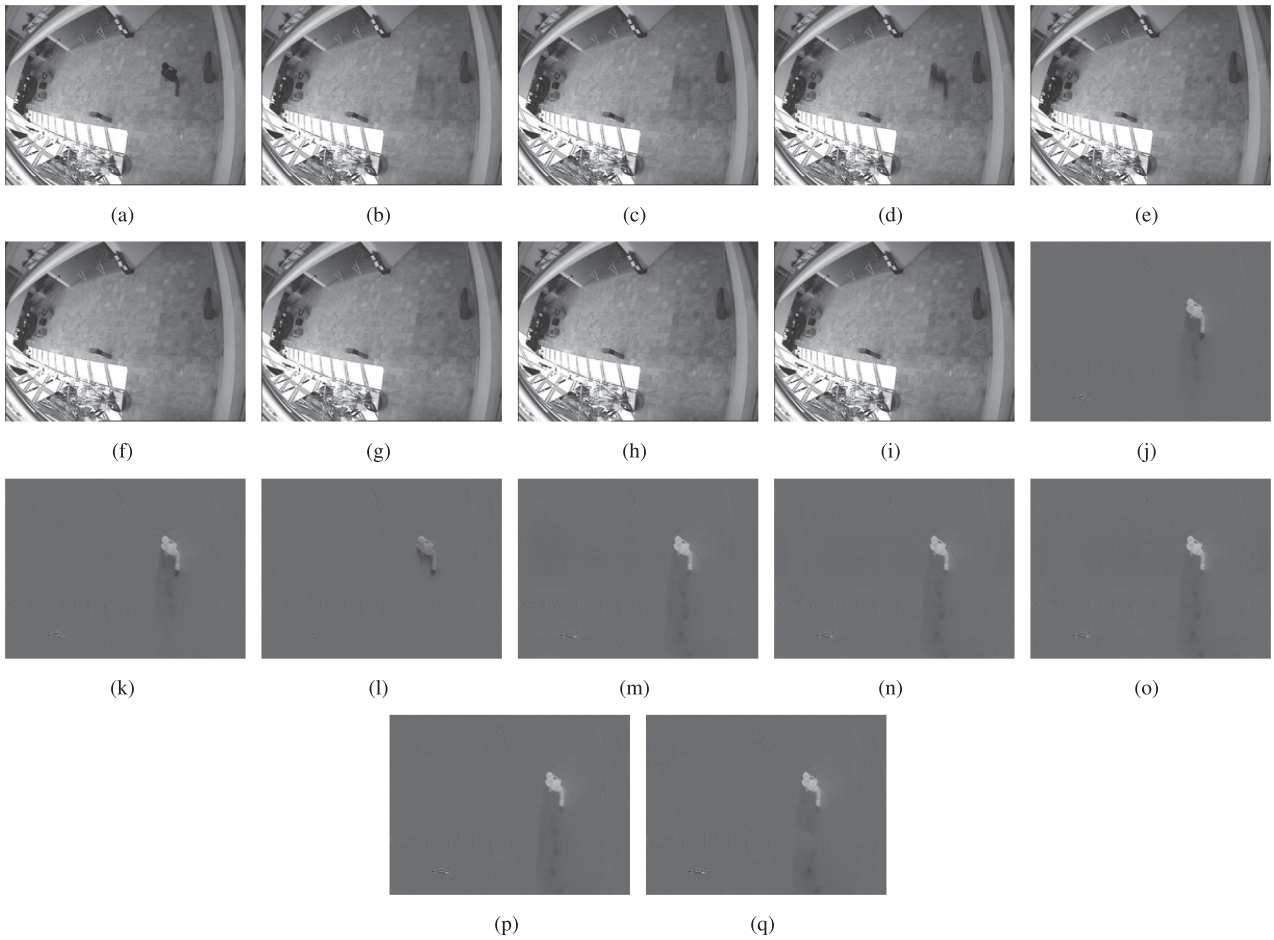



Fig. 14. Dynamic video foreground/background separation experiment. (a) Original 75-th frame (scene 1). Background extracted by (b) Adaptive Mean ($\lambda = 0.95$), (c) DTA ($\lambda = 0.95$), (d) DTA ($\lambda = 0.7$), (e) LRUT, (f) OSTD, (g) HOOI (increasing memory), (h) L1-HOOI (increasing memory), and (i) D-L1-TUCKER (proposed). Foreground extracted by (j) Adaptive Mean ($\lambda = 0.95$), (k) DTA ($\lambda = 0.95$), (l) DTA ($\lambda = 0.7$), (m) LRUT, (n) OSTD, (o) HOOI (increasing memory), (p) L1-HOOI (increasing memory), and (q) D-L1-TUCKER (proposed).

demonstrates better performance compared to that of $\lambda = 0.75$ at frame 75, however, at frame 150, we observe that DTA with $\lambda = 0.9$ captures some of the background from scene 1, while DTA with $\lambda = 0.7$ obtains a clean background and hence a smooth foreground, wherein the person appears slightly blurry. The proposed method is capable of tracking scene changes and we observe that it obtains a good estimate of the background and a clear foreground.

To quantify the background/foreground estimation performance, we compute, for every frame, the Peak Signal-to-Noise Ratio (PSNR) defined as $\text{PSNR} = 10\log_{10}(\frac{255^2}{\text{MSE}})$, where MSE is the mean squared error of the estimated background and the ground truth (clean) background. In Fig. 16, we plot the PSNR versus frame index and observe that all methods begin with high PSNR and, as they process frames with foreground movement, the PSNR drops. We observe that the PSNR of the proposed method is the highest after approximately frame 25. When the scene changes, the PSNR of all methods drops instantaneously. The PSNR values of HOOI, L1-HOOI, LRUT, and OSTD increase at a low rate as they process frames from the new scene. Adaptive mean and DTA with $\lambda = 0.95$ demonstrate better performance with faster PSNR increase. DTA with $\lambda = 0.5$ adapts to the new scene very quickly, but it is affected by foreground movement (depicted by oscillations in its PSNR values). The proposed method adapts to the new scene after it processes $W = 20$ measurements and attains the highest PSNR values across all frame indices thereafter. Certainly, after adaptation, the proposed method is straightforwardly capable of accurately extracting the background and foreground of all ambiguous frames in $\mathcal{W}$ in a retroactive fashion, as shown in Fig. 16.

### *E. Online Tucker and Classification*

In this experiment, we perform joint Tucker feature extraction and online classification. We use the Extended Yale Face Database B [58], consisting of face images of many individuals, captured at varying illumination. For this experiment, we use the face images of subject 02 and subject 23 to perform binary classification. Each class has 64 images in total, out of which, at every realization, we choose 45 for training and the remaining 19 for testing. The size of each image is originally $192 \times 168$ and

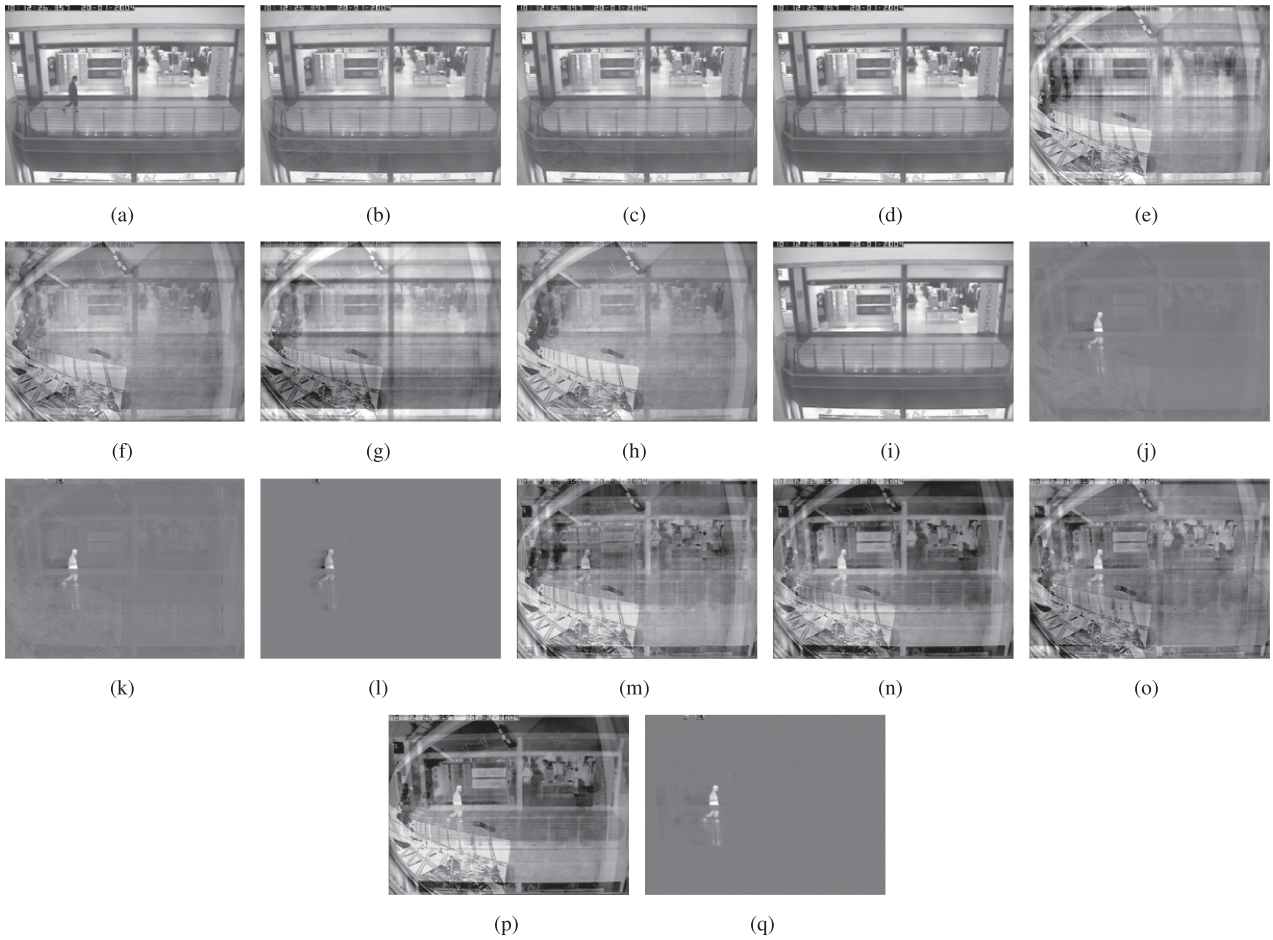



Fig. 15. Dynamic video foreground/background separation experiment. (a) Original 150-th frame (scene 2). Background extracted by (b) Adaptive Mean ($\lambda = 0.95$), (c) DTA ($\lambda = 0.95$), (d) DTA ($\lambda = 0.7$), (e) LRUT, (f) OSTD, (g) HOOI (increasing memory), (h) L1-HOOI (increasing memory), and (i) D-L1-TUCKER (proposed). Foreground extracted by (j) Adaptive Mean ($\lambda = 0.95$), (k) DTA ($\lambda = 0.95$), (l) DTA ($\lambda = 0.7$), (m) LRUT, (n) OSTD, (o) HOOI (increasing memory), (p) L1-HOOI (increasing memory), and (q) D-L1-TUCKER (proposed).

we down-sample it to $96 \times 84$. Therefore, at every realization, we have a tensor with training data $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{96\times 84\times 90}$ containing 90 measurements in total (45 from each class) and a tensor with testing data $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{96\times 84\times 38}$ containing 38 measurements in total (19 from each class). At every realization, we arbitrarily shuffle the training data and follow a parallel online feature extraction and classification approach as follows.

At update index $t$, we process the $t$-th training sample in $\boldsymbol{\mathcal{X}}$, $\bar{\boldsymbol{\mathcal{X}}}_t$, and update $\mathbf{Q}_1^{(t)}$ and $\mathbf{Q}_2^{(t)}$ using the proposed method ($B = 5,\ M = 10,\ d_1 = 15,\ d_2 = 6$, and cross-validated $\tau$). Next, we use the updated bases to compress all previously seen training data $\{\bar{\boldsymbol{\mathcal{X}}}_i\}_{i\in[t]}$ as $\boldsymbol{\mathcal{Z}}_i = \mathbf{Q}_1^{(t)\top} \bar{\boldsymbol{\mathcal{X}}}_i \mathbf{Q}_2^{(t)}$. We vectorize the compressed training measurements and give them as input to the Online Support Vector Machine (OSVM) classifier of [60].[4] We test the performance of the classifier on testing data, compressed using the same bases, and record the classification accuracy for every update index $t$. We repeat the experiment 300 times and plot the average classification accuracy versus update index in Fig. 17. Along with the proposed algorithm, we also plot the performance of the plain OSVM classifier, i.e., OSVM classifier run on vectorized (uncompressed) data, DTA with $\lambda = 0.33$, and OSTD. In Fig. 17, we observe that all compared methods attain almost identical performance. The classification accuracy starts low and as the update index increases it tends to 1.

Next, we repeat the experiment with the same setting, by corrupting each training measurement outside the initial memory batch $\mathcal{B}$ with noise from $\mathcal{N}(2, 5)$, cropping pixel intensities outside [0255]. We compute the average classification accuracy over 300 realizations and plot it versus update index in Fig. 17. In this case, we notice that plain OSVM is significantly affected by the noise. DTA, OSTD, and D-L1-Tucker demonstrate resistance to noise corruption, especially for earlier update indices.

### *F. Online Video Scene Change Detection*

In this experiment, we demonstrate the efficacy of the proposed method in online video scene change detection. We

[4]Matlab code available at https://www.cpdiehl.org/code.html.

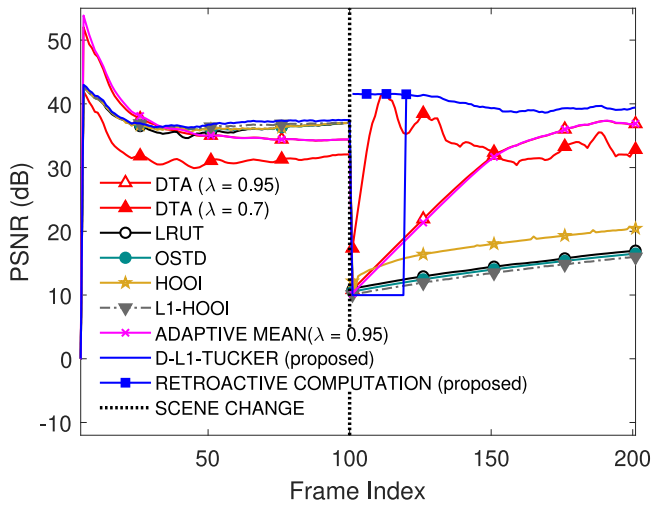



Fig. 16. Dynamic video foreground/background separation experiment. PSNR (dB) versus frame index.

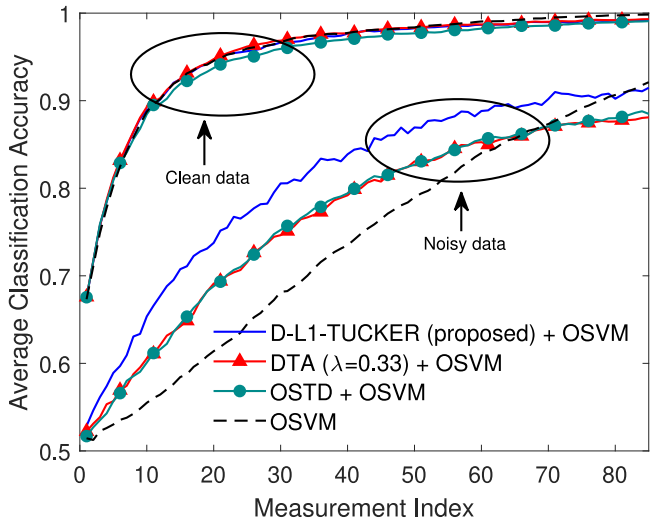



Fig. 17. Online tensor compression and classification experiment. Average classification accuracy versus update index.

operate on Red-Green-Blue (RGB) videos from the benchmark video scene change detection dataset in [59]. Specifically, we operate on three video frame-sequences from the dataset. Video 1 (`twoPositionPTZCam`) captures a street with a single scene change. Video 2 (`badminton`) captures badminton players in action with camera jittering. Video 3 (`zoomIn-zoomOut`) captures the backyard of a house with a single scene change. Each video can be seen as a collection of 3-way tensors $\bar{\boldsymbol{\mathcal{X}}}_t \in \mathbb{R}^{D_1 \times D_2 \times D_3}$ which are the video frames streaming across time $t = 1, 2, \ldots$. Videos 1, 2, and 3 are cropped to consist of $T = 202$, 220, and 172 frames, respectively, and each frame is of size $85 \times 143 \times 3$, $94 \times 144 \times 3$, and $60 \times 80 \times 3$, respectively. For videos 1 and 3, scene change occurs at frames 102 and 92, respectively.

We run the proposed algorithm on all three videos with $B = 20$, $M = 10$, $W = 5$, and cross-validated threshold $\tau$ set to 88% of the median reliability of the initialization batch. For all videos we set $d_3 = 1$. For videos 1 and 2 we set $d_1 = d_2 = 2$ while for video 3 we set $d_1 = d_2 = 5$.

A scene change is detected when the ambiguity batch overflows. The frame index of the first frame to enter the ambiguity batch is returned as the index of scene change. We quantify the performance of the proposed algorithm by means of the standard accuracy metric

$$\frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{TN} + \mathrm{FN}}, \tag{18}$$

where TP is the number of correct scene change detections, TN is the number of frames correctly identified as non-scene-changes, FP is the number of falsely identified scene changes, and FN is the number of missed detections. We compare the performance of the proposed algorithm with that of the state-of-the-art subspace-based scene change detection (SSCD) method in [61]. In addition, we extend SSCD to handle multi-way/tensor subspaces and replace the matrix products in Algorithm 1 of [61] by tensor products. Then, we apply SSCD on the tensor bases obtained by means of HOOI (batch), L1-HOOI (batch), and DTA ($\lambda = 0.2$). The values of $d_1$, $d_2$, and $d_3$ are the same as those used with the proposed algorithm. Other hyper-parameters of SSCD include positive constants $b$ and $c$ which are optimally tuned, individually for each method.

To evaluate the robustness of each method against corruptions, we corrupt each frame of all videos with probability $p_f$. To each pixel of a corrupted frame, we add salt-and-pepper noise with probability $p_n$. In Fig. 18, we illustrate a frame instance per scene for each video along with a noisy frame. For every value of $p_n$, we repeat the experiment 250 times.

For video 1 and $p_f = 0.1$, we illustrate the average detection accuracy of each method in Fig. 19 (left). We observe that under nominal conditions (no noise), the proposed algorithm and the tensor-based SSCD methods (HOOI + SSCD, L1-HOOI + SSCD, and DTA + SSCD), with $c = 10^7$ and $b = 1$ attain perfect performance by correctly identifying the scene change, without any false positives. In contrast, the plain SSCD method with $c = 10^{7.4}$ and $b = 1$ that relies on the $K = 2$ orthonormal bases obtained by SVD on the video frames demonstrates slightly degraded performance. This can be attributed to the loss of spatial context when the video frames are vectorized. As $p_n$ increases in steps of 0.005, we notice that plain SSCD is affected the most, followed by HOOI + SSCD and DTA + SSCD. L1-HOOI + SSCD exhibits some robustness to noise, comparatively. The proposed method maintains the best performance across the board.

For video 2 and $p_f = 0.25$, we report the average detection performance of each method in Fig. 19 (middle). The tensor-based SSCD methods use the same parameters as before while plain SSCD uses the same $K$ value, $c = 10^{7.5}$, and $b = 1$. Because this video is jittery, SSCD methods exhibit some robustness to noise for small $p_n$. As $p_n$ increases in steps of 1%, SSCD methods perceive camera jitter as scene change. In contrast, D-L1-Tucker remains robust to noise and is not misled.

Finally, for Video 3 and $p_f = 0.1$, we plot the detection performance of all methods in Fig. 19 (right). $p_n$ varies in steps of 0.01

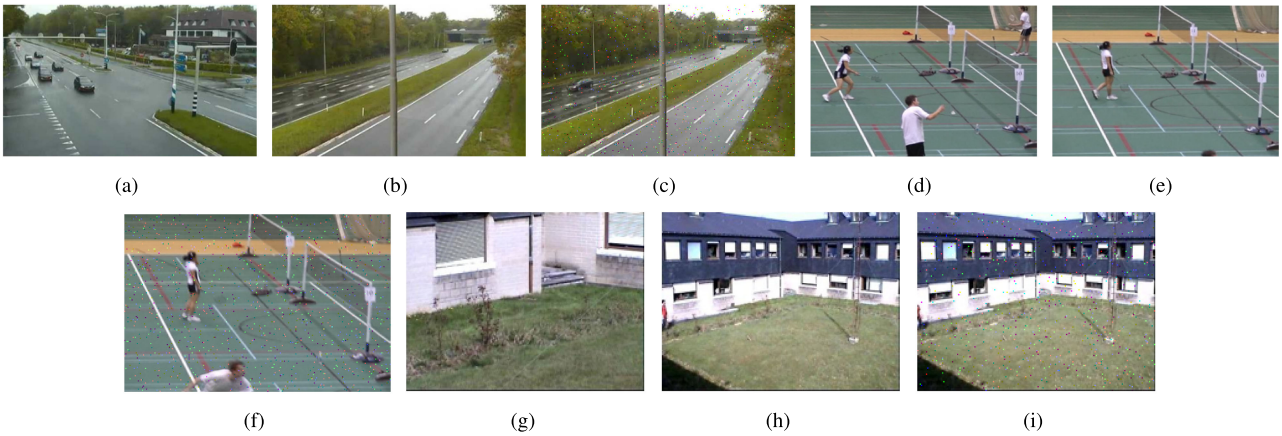



Fig. 18. Frame instances per scene for the three videos in [59]. Video 1: (a) scene 1, (b) scene 2, and (c) noisy frame. Video 2: (d) scene 1, (e) scene 2, and (f) noisy frame. Video 3: (g) scene 1, (h) scene 2, and (i) noisy frame. Probability of noise corruption per pixel is 10% for all noisy frames.

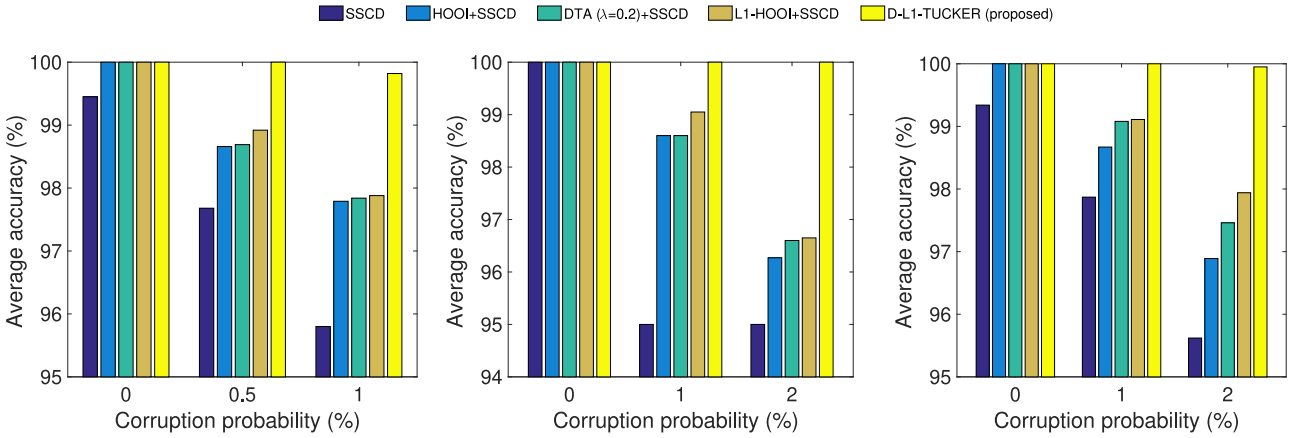



Fig. 19. Average online video scene change detection accuracy versus probability of noise corruption per pixel. For videos 1 and 3, the probability of frame corruption $p_f$ is set to 0.1 while for video 2 it is set to 0.25. Video 1 (left), video 2 (middle), and video 3 (right).

from 0 to 0.02. We use the same parameters for the tensor-based SSCD methods while for the plain SSCD method we use the same $K$, $c = 10^{7.25}$, and $b = 1$. We observe similar results to those of Fig. 19 (left) and Fig. 19 (middle). Although $p_n$ is small, the number of corrupted pixels per frame is significant –e.g., consider video 1 where each frame is of size $85 \times 143 \times 3$ and, on average, a probability of pixel corruption $p_n = 0.01$ results in approximately 365 noisy pixels per corrupted frame.

### G. Online Anomaly Detection

We consider the "Uber Pickups" tensor of the Formidable Repository of Open Sparse Tensors and Tools (FROSTT) [62] which is a $(N = 4)$-way tensor of size $D_1$-by-$D_2$-by-$D_3$-by-$D_4$ where $D_1 = 1140$ latitudes, $D_2 = 1717$ longitudes, $D_3 = 24$ hours, and $D_4 = 183$ days. Each entry of the tensor models number of Uber pickups in New York City over a period of about 6 months.

*Pre-processing:* To make the tensor more manageable in size, we first take a summation across the day mode and obtain a size $D_1$-by-$D_2$-by-$D_3$ tensor where $D_3 = 183$ days –i.e., a collection of 183 size 1140-by-1717 matrix measurements. We further reduce the size of the matrix measurements by retaining a 250-by-250 area centered at Manhattan wherein most of the activity –in terms of Uber Pickups– occurs. We consider the resulting tensor $\boldsymbol{\mathcal{X}}_{\text{uber}} \in \mathbb{R}^{250\times250\times183}$ to be a collection of 183 streaming measurements, one for each day.

*Streaming processing:* $\boldsymbol{\mathcal{X}}_{\text{uber}}$ can be seen as a data stream of matrix measurements each of which corresponds to a day. Accordingly, 7 successive measurements across the day index must correspond to a week which, in turn, is separated into weekdays and Saturdays. We assume that traffic during the weekdays is not the same as traffic on Saturdays and conjecture that weekdays belong to a coherent class/distribution while Saturdays belong to another.

We assume that we are given $B = 5$ measurements that correspond to weekdays and use those measurements to initialize D-L1-Tucker with memory size $M = 5$ and $d = 10$ components per mode. Moreover, we leverage these $B$ measurements to tune $\tau$ using the leave-one-out cross-validation approach that we presented above. We set $\tau$ to the median value of the $B$ collected reliability values in $\mathbf{r}$. Then, we update the decomposition of

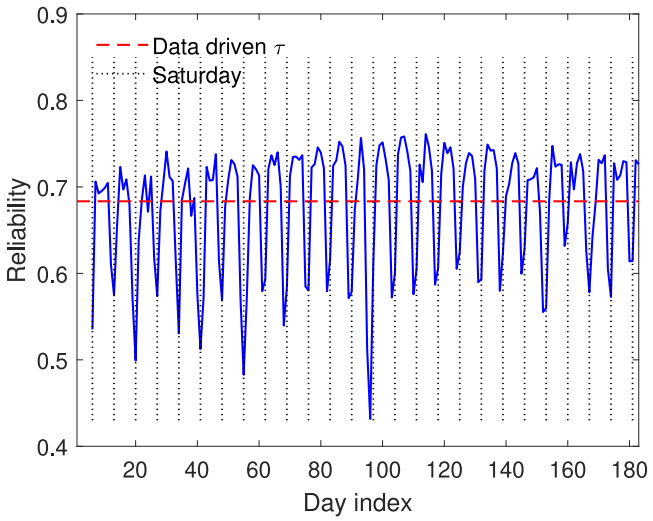



Fig. 20. $B = 5$, $M = 5$, $d = 10$, 300 realizations, data-driven $\tau$.

D-L1-Tucker by processing the rest of the measurements one by one. In Fig. 20, we plot the reliability of streaming measurements versus day (update) index. Moreover, we add the data-driven value of $\tau$ as a horizontal line. Each measurement with reliability value above this curve is deemed reliable for processing, while each measurement with reliability value below that curve is considered to be an anomaly (outlier). For better understanding, we include vertical dotted lines on the day indices which correspond to Saturdays. The reported curves are averaged over 300 random initializations of bases and runs of D-L1-Tucker on $\mathcal{X}_{\text{uber}}$. Quite consistently, days that correspond to Saturdays exhibit reliability values that are clearly below the $\tau$ threshold and are considered anomalies. In contrast, almost all weekdays exhibit reliability values above the $\tau$ threshold. A different selection of the threshold $\tau$ may slightly improve the results, however, even with the data-driven tuning of $\tau$, the reliability check feature of D-L1-Tucker offers high accuracy in identifying anomalies/outliers.

## VI. Conclusions

When tensor measurements arrive in a streaming fashion or are too many to jointly decompose, incremental Tucker analysis is preferred. In addition, dynamic bases adaptation is meaningful when the nominal data subspaces change. At the same time, outliers in the data can significantly compromise the performance of existing methods for dynamic Tucker analysis. In this work, we presented D-L1-Tucker: an algorithm for dynamic and outlier-resistant Tucker analysis of tensor data. Our experimental studies on real and synthetic datasets corroborate that the proposed method (i) attains high bases estimation performance, (ii) suppresses outliers, and (iii) adapts to changes of the nominal subspaces.

## References

[1] A. Cichocki *et al.*, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Process. Mag.*, vol. 32, no. 2, pp. 145–163, Mar. 2015.

[2] 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 Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, Jul. 2017.

[3] D. Tao, X. Li, W. Hu, S. Maybank, and X. Wu, "Supervised tensor learning," in *Proc. IEEE Int. Conf. Data Mining*, Houston, TX, Nov. 2005, p. 8.

[4] J. Kossaifi, Z. C. Lipton, A. Kolbeinsson, A. Khanna, T. Furlanello, and A. Anandkumar, "Tensor regression networks," *J. Mach. Learn. Res.*, vol. 21, no. 123, pp. 1–21, 2020.

[5] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Germany: Springer, 2006.

[6] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2012.

[7] H. Q. Ngo and E. G. Larsson, "EVD-based channel estimation in multicell multiuser MIMO systems with very large antenna arrays," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2012, pp. 3249–3252.

[8] M. B. Amin, W. Zirwas, and M. Haardt, "HOSVD-based denoising for improved channel prediction of weak massive MIMO channels," in *Proc. IEEE Veh. Technol. Conf.*, Jun. 2017, pp. 1–5.

[9] D. C. Araújo, A. L. F. de Almeida, J. P. C. L. Da Costa, and R. T. de Sousa, "Tensor-based channel estimation for massive MIMO-OFDM systems," *IEEE Access*, vol. 7, pp. 42133–42147, 2019.

[10] J. Yang, D. Zhang, A. F. Frangi, and J.-Y. Yang, "Two-dimensional PCA: A new approach to appearance-based face representation and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 1, pp. 131–137, Jan. 2004.

[11] R. Ballester-Ripoll, P. Lindstrom, and R. Pajarola, "TTHRESH: Tensor compression for multidimensional visual data," *IEEE Trans. Vis. Comput. Graph.*, pp. 2891–2903, Sep. 2019.

[12] H. Ben-Younes, R. Cadene, M. Cord, and N. Thome, "Mutan: Multimodal tucker fusion for visual question answering," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2631–2639.

[13] A. Koochakzadeh and P. Pal, "On canonical polyadic decomposition of overcomplete tensors of arbitrary even order," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process.*, Dec. 2017, pp. 1–5.

[14] E. E. Papalexakis, C. Faloutsos, and N. D. Sidiropoulos, "Tensors for data mining and data fusion: Models, applications, and scalable algorithms," *ACM Trans. Intell. Syst. Technol.*, vol. 8, pp. 16:1–16:44, Jan. 2017.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations*, *3 rd ed.* Baltimore, MD: Johns Hopkins Univ. Press, 1996.

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

[17] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, pp. 1253–1278, 2000.

[18] V. Barnett and T. Lewis, *Outliers in Statistical Data*., *2nd ed*. Hoboken, NJ, USA: Wiley, 1978.

[19] J. W. Tukey, "The future of data analysis," *Ann. Math. Statist.*, vol. 33, no. 1, pp. 1–67, 1962.

[20] X. Fu, K. Huang, W. K. Ma, N. D. Sidiropoulos, and R. Bro, "Joint tensor factorization and outlying slab suppression with applications," *IEEE Trans. Signal Process.*, vol. 63, no. 23, pp. 6315–6328, Dec. 2015.

[21] D. Goldfarb and Z. Qin, "Robust low-rank tensor recovery: Models and algorithms," *SIAM J. Matrix Anal. Appl.*, vol. 35, no. 1, pp. 225–253, 2014.

[22] P. P. Markopoulos, D. G. Chachlakis, and E. E. Papalexakis, "The exact solution to rank-1 L1-norm TUCKER2 decomposition," *IEEE Signal Process. Lett.*, vol. 25, no. 4, pp. 511–515, Jan. 2018.

[23] D. G. Chachlakis, A. Prater-Bennette, and P. P. Markopoulos, "L1-norm Tucker tensor decomposition," *IEEE Access*, vol. 7, pp. 178454–178465, 2019.

[24] Y. Pang, X. Li, and Y. Yuan, "Robust tensor analysis with L1-norm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 2, pp. 172–178, Feb. 2010.

[25] D. G. Chachlakis and P. P. Markopoulos, "Novel algorithms for exact and efficient L1-norm-based TUCKER2 decomposition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Apr. 2018, pp. 6294–6298.

[26] D. G. Chachlakis and P. P. Markopoulos, "Robust decomposition of 3-way tensors based on L1-norm," in *Proc. SPIE Defense Commercial Sens.*, Apr. 2018, pp. 1065807:1–1065807:15.

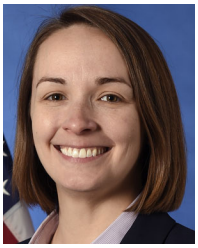
[27] X. Cao, X. Wei, Y. Han, and D. Lin, "Robust face clustering via tensor decomposition," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2546–2557, Nov. 2015.

[28] K. Tountas, D. G. Chachlakis, P. P. Markopoulos, and D. A. Pados, "Iteratively re-weighted L1-PCA of tensor data," in *Proc. IEEE Asilomar Conf. Signals, Syst. Comput.*, 2019, pp. 1658–1661.

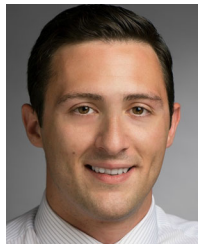
[29] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, and C. Faloutsos, "Incremental tensor analysis: Theory and applications," *ACM Trans. Knowl. Discov. Data*, vol. 2, no. 3, pp. 1–37, 2008.
[30] J. Sun, D. Tao, and C. Faloutsos, "Beyond streams and graphs: Dynamic tensor analysis," in *Proc. ACM Int. Conf. Knowl. Discov. Data Mining*, Aug. 2006, pp. 374–383.
[31] R. Yu, D. Cheng, and Y. Liu, "Accelerated online low rank tensor learning for multivariate spatiotemporal streams," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2015, pp. 238–247.
[32] P. P. Markopoulos, G. N. Karystinos, and D. A. Pados, "Optimal algorithms for L1-subspace signal processing," *IEEE Trans. Signal Process.*, vol. 62, no. 19, pp. 5046–5058, Oct. 2014.
[33] P. P. Markopoulos, D. G. Chachlakis, and A. Prater-Bennette, "L1-norm higher-order singular-value decomposition," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, Nov. 2018, pp. 1353–1357.
[34] D. G. Chachlakis, M. Dhanaraj, A. Prater-Bennette, and P. P. Markopoulos, "Options for multimodal classification based on L1-tucker decomposition," in *Proc. SPIE Defense Commercial Sens.*, Apr. 2019, pp. 109890O:1-109890O:13.
[35] D. G. Chachlakis, A. Prater-Bennette, and P. P. Markopoulos, "L1-norm higher-order orthogonal iterations for robust tensor analysis," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2020, pp. 4826–4830.
[36] P. P. Markopoulos, S. Kundu, S. Chamadia, and D. A. Pados, "Efficient L1-norm principal-component analysis via bit flipping," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4252–4264, Aug. 2017.
[37] S. Papadimitriou, J. Sun, and C. Faloutsos, "Streaming pattern discovery in multiple time-series," in *Proc. Int. Conf. Very Large Databases*, Aug. 2005, pp. 697–708.
[38] P. Narayanamurthy and N. Vaswani, "Provable dynamic robust PCA or robust subspace tracking," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2018, pp. 1–5.
[39] J. Feng, H. Xu, S. Mannor, and S. Yan, "Online PCA for contaminated data," in *Proc. Adv. Neural Inf. Process. Syst*, 2013, pp. 764–772.
[40] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1568–1575.
[41] P. P. Markopoulos, M. Dhanaraj, and A. Savakis, "Adaptive L1-norm principal-component analysis with online outlier rejection," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 6, pp. 1131–1143, Dec. 2018.
[42] Y. Liu, K. Tountas, D. A. Pados, S. N. Batalama, and M. J. Medley, "L1-subspace tracking for streaming data," *Pattern Recognit.*, vol. 97, pp. 106992:1–106992:13, 2020.
[43] M. Baskaran *et al.*, "Accelerated low-rank updates to tensor decompositions," in *Proc. IEEE High Perf. Extreme Comput. Conf.*, Sep. 2016, pp. 1–7.
[44] O. A. Malik and S. Becker, "Low-rank tucker decomposition of large tensors using tensorsketch," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10096–10106.
[45] R. Pagh, "Compressed matrix multiplication," *ACM Trans. Comput. Theory*, vol. 5, no. 3, pp. 1–17, 2013.
[46] Y. Sun, Y. Guo, C. Luo, J. Tropp, and M. Udell, "Low-rank tucker approximation of a tensor from streaming data," *SIAM J. Math. Data Sci.*, vol. 2, no. 4, pp. 1123–1150, 2020.
[47] 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," *Int. J. Comput. Vis.*, vol. 91, no. 3, pp. 303–327, 2011.
[48] X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo, "Robust visual tracking based on incremental tensor subspace learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.
[49] A. Sobral, C. G. Baker, T. Bouwmans, and E.-h. Zahzah, "Incremental and multi-feature tensor subspace learning applied for background modeling and subtraction," in *Proc. Int. Conf. Image Anal. Recognit.*, Oct. 2014, pp. 94–103.
[50] A. Sobral, S. Javed, S. Ki Jung, T. Bouwmans, and E.-h. Zahzah, "Online tensor decomposition for background subtraction in multispectral video sequences," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 106–113.
[51] S. Smith, K. Huang, N. D. Sidiropoulos, and G. Karypis, "Streaming tensor factorization for infinite data sources," in *Proc. SIAM Int. Conf. Data Mining*, May 2018, pp. 81–89.
[52] Y. Du, Y. Zheng, K.-C. Lee, and S. Zhe, "Probabilistic streaming tensor decomposition," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2018, pp. 99–108.
[53] M. Najafi, L. He, and S. Y. Philip, "Outlier-robust multi-aspect streaming tensor completion and factorization." in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 3187–3194.
[54] P. Li, J. Feng, X. Jin, L. Zhang, X. Xu, and S. Yan, "Online robust low-rank tensor modeling for streaming data analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 1061–1075, Apr. 2019.
[55] P. P. Markopoulos, S. Kundu, and D. A. Pados, "L1-fusion: Robust linear-time image recovery from few severely corrupted copies," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2015, pp. 1225–1229.
[56] K. Tountas, G. Sklivanitis, D. A. Pados, and M. J. Medley, "Tensor data conformity evaluation for interference-resistant localization," in *Proc. IEEE Asilomar Conf. Signals, Syst. Comput.*, 2019, pp. 1582–1586.
[57] *Context aware vision using image-based active recognition (CAVIAR)*. Accessed: Jul. 2020. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/
[58] The extended yale face database B. Accessed: Jul. 2020. [Online]. Available: http://vision.ucsd.edu/iskwak/ExtYaleDatabase/ExtYaleB.html
[59] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2012, pp. 1–8.
[60] C. P. Diehl and G. Cauwenberghs, "SVM incremental learning, adaptation and optimization," in *Proc. IEEE Int. Joint Conf. Neural Net.*, Jul. 2003, pp. 2685–2690.
[61] Y. Jiao, Y. Chen, and Y. Gu, "Subspace change-point detection: A new model and solution," *IEEE J. Select. Top. Signal Process.*, vol. 12, no. 6, pp. 1224–1239, Oct. 2018.
[62] S. Smith *et al.*, "FROSTT: The formidable repository of open sparse tensors and tools," 2017. [Online]. Available: http://frostt.io/

**Dimitris G. Chachlakis** (Member, IEEE) was born in Athens, Greece, in 1991. He received the Diploma (five-year program) degree in electronic and computer engineering from the Technical University of Crete, Chania, Greece, in 2016. Then, he started working toward the Ph.D. degree with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, New York, USA. He was a Research Assistant with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, New York, USA. His research interests include the areas of machine learning, signal processing, and data analysis, with a focus on robustness, tensor methods, and sparse array processing. He is a Member of the IEEE Signal Processing Society and the SIAM. He was a Reviewer to the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE SIGNAL PROCESSING LETTERS, the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE ACCESS, the IEEE PHOTONICS JOURNAL, and the *ELSEVIER Digital Signal Processing*. He was the recipient of the 2018 Gerondelis Foundation Graduate Student Scholarship Award and the 2019 A. G. Leventis Foundation Graduate Student Scholarship Award.

**Mayur Dhanaraj** (Student Member, IEEE) was born in Bengaluru, India, in 1994. He received the B.S. degree in electronics and communications engineering from the Bangalore Institute of Technology, Bangalore, India (affiliated to Visvesvaraya Technological University, Belgaum, India), in 2016, and the M.S. degree in electrical and microelectronic engineering from the Rochester Institute of Technology (RIT), Rochester, NY, USA, in 2018. Since September 2018, he has been working toward the Ph.D. degree with the Kate Gleason College of Engineering, RIT. He is currently a Research Assistant with the Kate Gleason College of Engineering, RIT. His research interests include the areas of machine learning, signal processing, and reliable data analysis.



**Ashley Prater-Bennette** (Member, IEEE) received the Ph.D. degree in mathematics from Syracuse University, Syracuse, NY, USA, in 2013. She is currently a Senior Research Mathematician with Air Force Research Laboratory, Information Directorate in Rome, NY, USA. She currently leads the applied research portfolio in complex effect analysis within the autonomy, command and control core technical competency for the Air Force and is currently the Machine Intelligence Technical Area Lead for in-house research efforts within the Information Directorate. Her research interests include mathematical foundations of machine learning, tensor decompositions, and sparse and low-rank data representations.



**Panos P. Markopoulos** (Member, IEEE) was born in Athens, Greece, in 1986. He received the Diploma (five-year program) degree in electronic and computer engineering, the M.S. degree in electronic and computer engineering, from the Technical University of Crete, Chania, Greece, in 2010 and 2012, respectively, and the Ph.D. degree in electrical engineering from The State University of New York at Buffalo, Buffalo, NY, USA, in 2015. Since August 2015, he has been an Assistant Professor of electrical engineering with the Rochester Institute of Technology (RIT), Rochester, NY, USA, where he directs the Machine Learning Optimization and Signal Processing Laboratory. He is also a Core Faculty with the RIT Center for Human-Aware Artificial Intelligence. In 2018 and 2020, he was a Summer Visiting Research Faculty with the U.S. Air Force Research Laboratory, Information Directorate, in Rome, NY, USA. He has coauthored more than 65 journal and conference articles in areas of his research interests, which include statistical signal processing, machine learning, data analysis, and optimization, with a current focus on tensor methods, robustness, Lp-norm formulations, and dynamic learning. He was the Principal Investigator of multiple research projects funded by the U.S. National Science Foundation, the U.S. National Geospatial-Intelligence Agency, and the U.S. Air Force Research Laboratory. In 2020, he was the recipient of the the prestigious AFOSR Young Investigator Award. He is a Member of the IEEE Signal Processing, Computer, and Communications Societies, with high service activity that includes the organization of multiple conference events, such as the 2019 IEEE International Workshop on Machine Learning for Signal Processing, the 2019–2021 versions of the SPIE DCS Conference on Big Data: Learning Analytics and Applications, and the 2017–2020 versions of the IEEE International Workshop on Wireless Communications and Networking in Extreme Environments.