# Multi-aspect-streaming tensor analysis

**Article** *in* Knowledge-Based Systems · November 2015

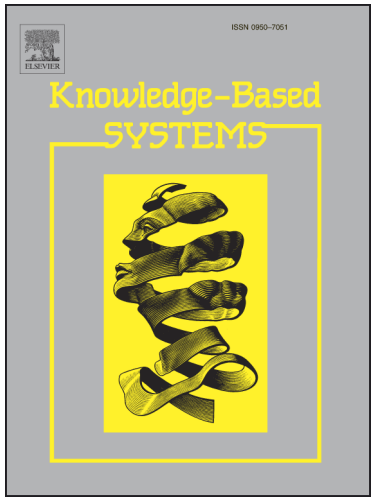
DOI: 10.1016/j.knosys.2015.07.013



**2 authors:**

Hadi Fanaee-T
University of Porto
**18** PUBLICATIONS **47** CITATIONS
SEE PROFILE

João Gama
University of Porto
**325** PUBLICATIONS **4,000** CITATIONS
SEE PROFILE




# Multi-aspect-streaming tensor analysis

Hadi Fanaee-T, Joao Gama

PII: S0950-7051(15)00267-1
DOI: http://dx.doi.org/10.1016/j.knosys.2015.07.013
Reference: KNOSYS 3214

To appear in: *Knowledge-Based Systems*

Received Date: 12 January 2015
Revised Date: 14 July 2015
Accepted Date: 15 July 2015

Please cite this article as: H. Fanaee-T, J. Gama, Multi-aspect-streaming tensor analysis, *Knowledge-Based Systems* (2015), doi: http://dx.doi.org/10.1016/j.knosys.2015.07.013

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Knowledge-Based Systems 00 (2015) 1–21

Knowledge-Based Systems

# Multi-aspect-streaming tensor analysis

Hadi Fanaee-T[1], Joao Gama[b]

[a]*Laboratory of Artificial Intelligence and Decision Support, FCUP/University of Porto, Rua Dr. Roberto Frias, 4200 - 465 Porto, Portugal*
[b]*Laboratory of Artificial Intelligence and Decision Support, FEP/University of Porto, Rua Dr. Roberto Frias, 4200 - 465 Porto, Portugal*

**Abstract**

Tensor analysis is a powerful tool for multiway problems in data mining, signal processing, pattern recognition and many other areas. Nowadays, the most important challenges in tensor analysis are efficiency and adaptability. Still, the majority of techniques are not scalable or not applicable in streaming settings. One of the promising frameworks that simultaneously addresses these two issues is Incremental Tensor Analysis (ITA) that includes three variants called Dynamic tensor analysis (DTA), Streaming tensor analysis (STA) and Window-based tensor analysis (WTA). However, ITA restricts the tensor's growth only in time, which is a huge constraint in scalability and adaptability of other modes. We propose a new approach called multi-aspect-streaming tensor analysis (MASTA) that relaxes this constraint and allows the tensor to concurrently evolve through all modes. The new approach, which is developed for analysis-only purposes, instead of relying on expensive linear algebra techniques is founded on the histogram approximation concept. This consequently brought simplicity, adaptability, efficiency and flexibility to the tensor analysis task. The empirical evaluation on various data sets from several domains reveals that MASTA is a potential technique with a competitive value against ITA algorithms.



*Keywords:*
Tensor analysis, data streams, online histogram, tensor decomposition, streaming tensor analysis

## 1. Introduction

Tensor decomposition is a powerful technique for the analysis of multiway data in psychometrics, chemometrics, network information systems, pattern recognition and data mining [1]. The growing interest in tensors is due to their capability of discovering complicated patterns in multiway settings that is impossible via other methods. Many techniques are developed for tensor decomposition, but two of the most popular ones are Tucker [2] and PARAFAC [3]. Both of these models suffer from two major issues. Firstly, they are not scalable to large size data sets due to their time/space complexity; and secondly, are not updatable when a new stream of data is retrieved.

The scalability issue is already addressed in three major groups of solutions, including sparse-optimized methods, parallel and distributed techniques and GPU-based solutions. For instance, in [4] a new extension of Tucker decomposition is proposed, called Memory-efficient Tucker (MET) that its space complexity scales up to the non-zero elements in tensor (i.e. O(nz)). In [5] a distributed version of PARAFAC is implemented in MapReduce [6] scaling PARAFAC decomposition up to 100 times for sparse tensors. A different distributed framework is proposed in [7, 8] for PARAFAC that divides the tensors into some small sub-tensors and solve sub-tensors problems in different machines. Similar to these works, [9] proposes a parallelized version of PARAFAC called ParCube which is

*Email address:* hadi.fanaee@fe.up.pt (Hadi Fanaee-T)

optimized for sparse tensors and provides 14 times acceleration in runtime. In [10] a new method is proposed based on general-purpose computing, on the GPU that operates 360 times faster than the regular PARAFAC decomposition.

Although the above techniques are great tools for dealing with large tensors, they suffer from the non-adaptability problem. This means that when new data is received we have to rebuild the model from scratch. In addition, sparsity-optimized techniques such as MET also do not have any added value for dense tensors, because they only scale up when there is a considerable amount of zero elements in the tensor. Furthermore, the parallelization of tensor decompositions is not as straightforward and it requires extra hardware and software infrastructures.

The pioneer research studies on this problem are those performed by [11, 12, 13] who propose some streaming approximation solutions for tensor decomposition in an unified framework called incremental tensor analysis (ITA). The ITA solution, opposed to other scalable decomposition techniques does not need any special infrastructure. It also does not make any restrictive assumption like sparsity. It performs tensor decomposition on each tensor in each time instant, maintains some statistics and then incorporates that for the processing of the next tensor. Therefore, it does not require keeping historical data in the memory. This solution has two advantages. First, tensor model is easily updatable when new data arrives, and second, the space required for decomposition of the tensor becomes independent of stream length.

The merits of ITA and its usefulness to the analysis of time-evolving tensors are investigated in many studies, so that nowadays, ITA is recognized as the state-of-the-art solution for streaming tensor analysis. However, although ITA allows the tensor to evolve infinitely in time, it makes a restrictive assumption that the dimension of the tensor remains constant during the process. We may not find this limitation annoying for only-time-evolving tensors like network traffic or video streams, when the number of nodes or image frames remain constant during the analysis. But, we may deeply feel this constraint in dealing with multi-aspect-evolving tensors such as social networks, where the number of nodes grows during the evolution of the network. Or in recommendation systems when new users are joined to the system, and size of $user \times profile$ matrix consistently changes. Aside from that, ITA encounters the *intermediate data explosion problem* [5, 14] as well as its offline counterparts when the size of the tensor is large.

The intermediate data explosion problem corresponds to the heart of the these techniques, i.e. space-inefficient linear algebra computations that operate directly on the input data. Therefore, in these methods, space efficiency is more influenced by the size of input data rather than the method per se. However, we know that a large portion of tensor decomposition applications is related to *analysis-only* tasks such as anomaly detection (e.g. [15, 16, 17, 18]) or simple data analysis (e.g. [19, 20, 21, 22]). In such applications, computing the exact subspace of the tensor may not seem mandatory, as opposed to other applications such as compression where the reconstruction of tensor is inevitable. Can we find an alternative adaptive solution for tensor analysis that on one hand avoids space-inefficient computations and on the other hand provides the basic analytical power of tensor decomposition?

We know that histograms are central tools for summarization in data mining. They are also the key technique in image retrieval for measuring similarity between images. Is it possible to extend these ideas to tensor analysis problem? We may find a positive answer for this question, but two more questions will be raised in the following: a) how do we deal with the huge space/time complexity of histograms while we actually require an efficient method?; b) is it conceivable to utilize a non-adaptive tool like histogram for solving a streaming problem?

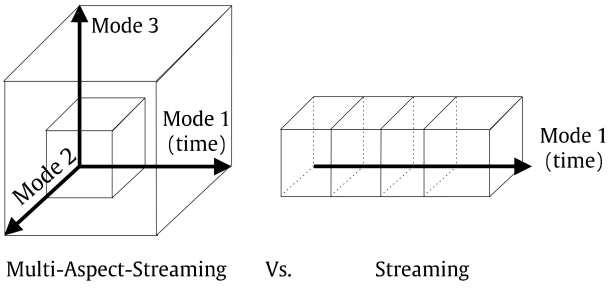



Figure 1: Comparison of multi-aspect-streaming tensor analysis (proposed) versus streaming tensor analysis (state-of-the-art).

In this research, we tackle these problems by recommending a histogram-based solution that allows the tensor to simultaneously evolve through all modes. We initiate with the description of fundamental concepts such as histograms, tensor segmentation and distribution matching and proceed to develop the first basic approach for histogram-based tensor analysis. Furthermore, we extend the baseline solution to the multi-aspect-streaming scheme (see Fig. 1 ) by replacing the conventional histogram with a recent incremental approach. To the best of our knowledge the application of histograms in tensor analysis is not reported elsewhere. This is also the first work that addresses the multi-aspect-streaming tensor analysis problem.

The rest of the paper is organized as follows: Section 2 outlines the preliminary concepts. In section 3 we describe the proposed method. We introduce a new evaluation methodology in section 4 and later employ it for assessment of the proposal in section 5. Next, in section 6 we illustrate the application of the proposed approach on two real case studies. The last section concludes the exposition, presenting the final remarks.

## 2. Preliminary concepts

Following [23], throughout the paper, scalars are denoted by non-bold lowercase letters (e.g. $i$), vectors are denoted by boldface lowercase letters (e.g. $\boldsymbol{a}$), matrices are denoted by boldface capital letters (e.g. $\boldsymbol{A}$) and tensors are denoted by Calligraphic letters (e.g. $\boldsymbol{\mathcal{X}}$). In the following we define the necessary concepts required for further description of the proposed methodology. More comprehensive discussion about tensors and their application can be found in survey papers [23, 1].

### 2.1. Tensor

A tensor is a multi-dimensional array and the order of a tensor is the number of dimensions, also known as ways or modes. Vectors, matrices and tensors respectively, are equivalent to first, second and $d$th order tensor where $d \geq 3$.

### 2.2. Slice

A slice is a *(d-1)*-dimension partition of tensor when an index is fixed in one mode and the indices vary in the other modes. The horizontal, lateral, and frontal slides of a third-order tensor $\boldsymbol{\mathcal{X}}$, are denoted by $X_{i::}$, $X_{:j:}$, and $X_{::k}$, respectively. Each slice in each mode corresponds to an entity (or feature). For instance, in a three-order tensor of *country* $\times$ *year* $\times$ *measurement*, the country "Portugal" is a feature in the first mode. The year 2014 is an entity in the second mode and "population" or "GDP" are the features in the third mode.

## 3. Histogram-based tensor analysis

Histograms are simple statistical tools that have been applied in a wide range of applications [24]. They are simple, non-parametric and easy to interpret, which make them attractive for summarizing of data. Histograms are extensively used in the mining and processing of data streams [25, 26, 27] to keep the abstract of past data; in database management systems for cost estimation in query optimization [28, 29]; and in image retrieval [30, 31] for image matching. With some inspirations from these applications, we intend to extend the application of histograms to the tensor analysis problem. In the following paragraph we explain the logic for this selection.

We know that a $d$-dimensional tensor is composed of multiple *(d-1)*-dimensional *slices* in each mode. For a 3D tensor as is depicted in Fig. 2, slices are 2D matrices. Each slice contains a particular segment of the tensor information, so that if all slices get combined together they rebuild the original tensor. In tensor analysis, we assume that many of the features are correlated with others and they jointly explain the data. From image matching application, we know that histograms are useful for measuring the similarities between the two images. Hence, it is rational to assume:

**Assumption 1** *If two features are similar the histogram of their corresponding slices should be similar as well.*

However, we may have two slices with totally different correlation patterns that have similar histograms. For instance, in image matching we may find two different images with similar histograms [31]. Therefore, this assumption might be violated in some applications. However, we presume that this is not the case in the majority of applications.

Aside from this, measuring distribution distances between all pairs of slices is an exhaustive task. We believe that if two slices have similar distances to the tensor distribution (as a reference), they probably are similar because they explain the same part of tensor information. Therefore, maybe it is better to instead of performing exhaustive match between all pairs of slices, only compute the similarity of slice histograms to the tensor histogram. However, this needs to be assumed:

**Assumption 2** *Two features are similar, if histograms of their corresponding slices have a similar distance to the tensor histogram.*

With the new assumption, anomalous slices (or features) are also easy to discover. Those slices that have a totally different histogram when comparing to other slices are considered abnormal, so if we remove them from the tensor, we still can explain the majority of information using the remaining slices.

Using the above assumptions, we propose the first histogram-based method for tensor analysis. In the following subsections we first present the detailed presentation of the basic algorithm and then proceed with the introduction of its multi-aspect-streaming extension.

### *3.1. Offline histogram-based tensor analysis (OHTA)*

#### *3.1.1. Theory*

Let us denote the tensor data with $\boldsymbol{\mathcal{X}}$ and each slice of tensor with $\mathbf{X}_{d,m}$ where m is denotes the m*th* slice in mode d. Also let us denote the vectorized form of $\boldsymbol{\mathcal{X}}$ and $\mathbf{X}_{d,m}$ respectively with $\mathbf{x}_r$ and $\mathbf{x}_{d,m}$. Assuming k as the number of bins, the histograms of $\mathbf{x}_r$ and $\mathbf{x}_{d,m}$ respectively will be $P_k(x_r)$ and $P_k(x_{d,m})$. As each group of slices represent a different part of information in $\boldsymbol{\mathcal{X}}$ it is expected that histogram of slices in the same cluster have similar distances to the tensor histogram, $P_k(x_r)$. This concept is frequently used in the real-life. For example in sea-floor mapping, one approach to estimate ocean depth is emitting a sound wave to the water and record the time it takes for sound wave to be reflected back. The surfaces with same depth reflect sound with same speed, hence will be clustered together as same-level. Likewise, pits and holes are identified easily since they reflect sound with totally different speed comparing other surfaces.

Based on a similar idea we compute the distance between $P_k(x_{d,m})$ and the $P_k(x_r)$ with Earth Mover Distance (EMD) [32]. EMD is widely used in image retrieval for computing distances between the color histograms of two images. We may want to use other distance measures, but EMD is the only one that is suitable for partial match purposes [32], hence it seems more appropriate here. The distribution of distances tells us what groups of slices are similar and which are not. If we again build another histogram on the vector of obtained distances we can cluster the slices to $k$ groups. Naturally, slices that are located in lower frequency bins can be marked as abnormal. As each slice corresponds to a specific feature, we can identify abnormal features in each mode.

#### *3.1.2. Algorithm*

In this section we introduce OHTA, a baseline algorithm for histogram-based tensor analysis. This method which is presented in Algorithm 1 requires three inputs: tensor $\boldsymbol{\mathcal{X}}$, number of bins for reference histogram $b1$ and number of bins for distances histogram $b2$. The $b2$ parameter is somehow equivalent to the number of components in PARAFAC decomposition in the sense that OHTA finds a $b2$ group of features in each mode.

The algorithm initiates with vectorization of the tensor $\boldsymbol{\mathcal{X}}$ where $d$-dimensional tensor is transformed into an one-dimensional vector $\boldsymbol{x}$ (Fig. 2-1) and then its histogram which is called *reference histogram* (or simply tensor histogram) is calculated with $b1$ numbers of bins. Next, for each slice in each mode (Fig. 2-2) we compute its histogram according to the bins obtained for the reference histogram (Fig. 2-3). Note that, the way we generate histograms for slices is different from when we apply the histogram directly to the slices. Here, for all slices we use the same bins as the reference histogram. For instance in Fig 2, if the bins for the reference histogram are (25, 20, 15, 10, 5) and vectorized slice 1,1 is (26, 24, 19, 27, 14, 16, 10, 4, 3), the histogram of slice 1,1 will be (25:3, 20:1, 15:2, 10:1, 5:2). This is obtained as follows. We first create an empty copy of the reference histogram (25:0, 20:0, 15:0, 10:0, 5:0). Then for each quantity in the slice we add one to the count corresponding to the closest point. For instance, for 26 the closest bin in the reference histogram is 25. For 24 and 27 the closest point is also 25. Therefore, bin 25 gets frequency count of 3. For bin 20, the count is 1 because among all values only one item, i.e. 19 has been the

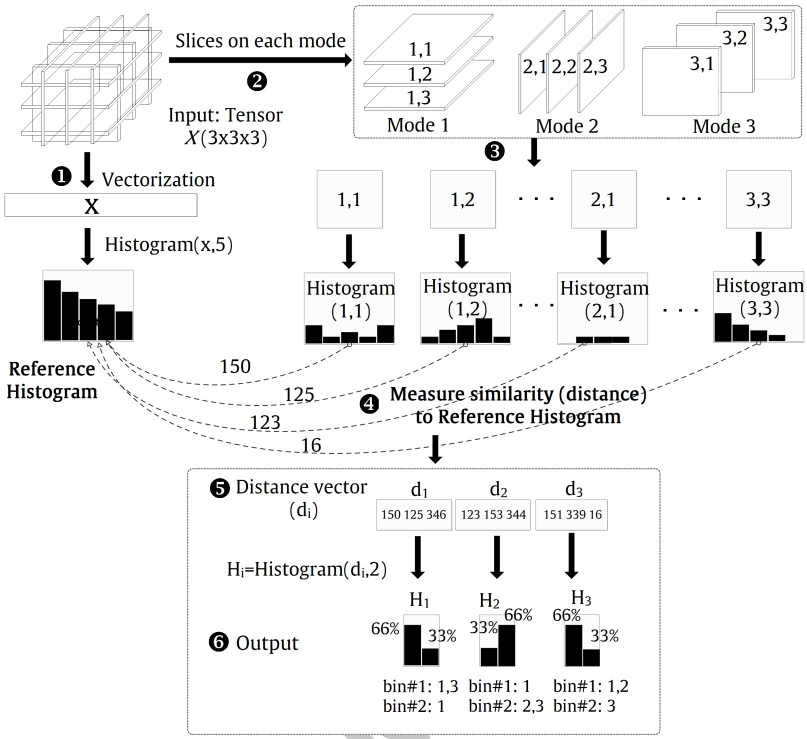



Figure 2: An illustrative example of offline histogram-based tensor analysis (OHTA): 1) a 3D tensor $\boldsymbol{X}(3 \times 3 \times 3)$ is transformed to a vector $\boldsymbol{x}$ and its corresponding histogram (called reference histogram) is obtained with $b1 = 5$ number of bins; 2) The horizontal, lateral and frontal slices of the tensor are obtained; 3) All slices from all modes are vectorized and their histograms are calculated using bins obtained for reference histogram; 4,5) EMD distance of each slice histogram to the reference histogram is calculated and is kept in vector of distance for each mode ($d_i$); 6) histogram ($d_i$,b2=2) is calculated for each mode $i$ and the member indices and the corresponding support are reported.

closest point to 20. We do not count 24 for bin 20 because 24 is closer to 25 than 20. Via this procedure, we generate the histograms for all slices.

In the next step, we compute the EMD distance of the slice histograms to the reference histogram (Fig. 2-4) and save the distances for each mode in $\boldsymbol{d}_i$ vector (Fig. 2-5).

We finally compute the histogram of EMD distances, $\boldsymbol{d}_i$ for mode $i$ with $b2$ numbers of bins (Fig. 2-6). The constructed histogram, $H_i$ is the output of OHTA. Each bin in $H_i$ represents a cluster of features and the cluster support is equal to frequency count of the histogram.

### *3.2. Multi-aspect-streaming tensor analysis (MASTA)*

#### *3.2.1. Theory*

Although histograms are very useful tools for summarization of data, they have two problems that make them impractical for streaming or large-scale data analysis. Firstly, they are computationally very expensive; and secondly it is not possible to update the histogram when new data arrives.

Fortunately, advances in data stream mining and database management systems has brought new efficient techniques for histogram approximation. There exist two group of techniques for histogram approximation. The first group are those non-constant space approaches that offer approximation guarantee. For instance, the proposed approach by [33] requires space of $O(logn)$. The second group (which we exploit in our solution) includes those methods that practically present a good performance and consume lower space. For instance, the recent algorithm by [34] is

**Algorithm 1** OHTA

**Require:** Tensor $\boldsymbol{\mathcal{X}}$ , bins in reference histogram ($b1$), bins in distances histogram ($b2$)
**Ensure:** $H_i$
1: Transform Tensor $\boldsymbol{\mathcal{X}}$ to vector $\boldsymbol{x}$
2: Reference Histogram ← Histogram ($\boldsymbol{x}, b1$)
3: **for** each mode i **do**
4: **for** each slices **do**
5: Create histogram of the slice according the bin centers obtained for reference histogram
6: $\boldsymbol{d}_i \xleftarrow{add}$ EMD(slice histogram, reference histogram)
7: **end for**
8: $H_i$ ← Histogram ($\boldsymbol{d}_i, b2$)
9: **end for**

**Algorithm 2** MASTA Update

**Require:** data chunk $\boldsymbol{s}$, $b1$, old reference histogram, old slice histograms for each mode
**Ensure:** reference histogram, slice histograms for each mode
1: Update reference histogram with respect to elements in $\boldsymbol{s}$, b1 and old reference histogram
2: Update or add histogram of slices corresponding to modes of $\boldsymbol{s}$, old slice histogram and b1

constant-space approach. However, The main problem about these approaches is that they lack any rigorous and accurate analysis [34]. On the other side, non-constant space approaches (i.e. first category) can be quite problematic for analyzing large-scale data. Nevertheless, the constant-space methods come with some restrictions that should carefully be taken into account. For instance, as stated by the authors in [34], when the data distribution is largely skewed, we should not expect a good approximation from these approaches. However, good accuracy might be expected when we deal with categorical distributions with a limited number of values. This adequately matches the condition we usually encounter in tensor analysis.

The theory part of MASTA is the same as OHTA, so we do not repeat it once more. The major difference is that in MASTA we replace the offline histogram with the online histogram approximation algorithm [34]. There are some other minor differences that will be outlined later in the algorithm explanation section. The detailed explanation of the online histogram approximation algorithm is out of the scope of this paper. The readers are referred to section 2.1 [34] for more details about the online histogram algorithm. However, implementation of the algorithm is very straightforward. Instead of keeping the whole data, we update the old histogram through the following procedures: update, merge, sum and uniform. The idea is that given a determined maximum number of bins when new data item arrives, if its value is close to each of the previous bins it is allocated to that bin and the corresponding bin center is updated accordingly. However, if the new data item is far away from the previous bins, a new bin is created and two of the closer bins are merged. This process continues until approximation of the whole data histogram.

Employment of an online histogram in OHTA accomplishes two functions: first, it promises a huge space efficiency, because space complexity of the online algorithm is O(1) versus the expensive offline approach O(N); and second, we do not face the intermediate data explosion problem [5, 14]. Because, each piece of data upon arrival enters to the model, updates the model and then is removed from the memory. This piece of data should not necessarily be a tensor or matrix as existing tensor solutions, rather it can be a single element of the tensor.

*3.2.2. Algorithm*

In this section we present the multi-aspect-streaming edition of OHTA, called MASTA by replacing the exact histogram calculation in OHTA with the above-mentioned online method. Some other modifications are also required to be taken into account that will be explained in the following. MASTA is composed of three procedures called update (Algorithm 2), output (Algorithm 3) and slice reconstruction (Algorithm 4).

Algorithm 2 presents the update procedure. This process is responsible for updating the model upon data arrival. In this algorithm we update the reference histogram and slice histograms in all modes upon new stream arrives. We also create the histogram if it does not exist. A toy example of this process is illustrated in Fig. 3.

**Algorithm 3** MASTA Output

**Require:** reference histogram, slice histograms for each mode, $b2$
**Ensure:** $H_i$
1: **for** each mode $i$ **do**
2: **for** each slice $j$ **do**
3: $R_j \leftarrow$ Compute reconstructed slice histogram using Algorithm 4
4: $\boldsymbol{d}_i \overset{add}{\longleftarrow}$ EMD($R_j$, reference histogram)
5: $H_i \leftarrow$ Update Histogram ($\boldsymbol{d}_i$, $b2$).
6: **end for**
7: **end for**

**Algorithm 4** MASTA Slice histogram reconstruction

**Require:** reference histogram, slice histogram, b1
**Ensure:** R
1: R ← copy reference histogram with frequency counts of zero
2: **for** each bin in R **do**
3: Find the closest point in slice histogram
4: Add the frequency count corresponding to the found point to the current bin's count.
5: **end for**

Algorithm 3 is executed when the user asks for the model result. The inputs of this algorithm are the outputs of Algorithm 2. In the output procedure, the first step is rebuilding the slice histogram with respect to the reference histogram via Algorithm 4. The reconstruction process in Algorithm 4 works as follows: we create an empty histogram with same bins as reference histogram and then instead of allocating the original counts in reference histogram we calculate the counts from the slice histograms. Via this procedure, we build histograms for all slices. After that, as OHTA we compute the EMD distance of each slice histogram to the reference histogram. We keep distances for all slices in each mode $d_i$. Finally, similar to OHTA we compute the online histogram of distance vectors, $d_i$ and report the slice indices and the corresponding frequency count for each bin.

## 4. Experimental evaluation

An ideal tensor analysis approach is the one that presents the most accurate model while it uses less resources (time and space). Therefore, as well as many learning algorithms, usefulness of any approximation solution should be evaluated based on a trade-off between accuracy and efficiency. In this section, we introduce the datasets, experimental settings, and the evaluation strategy we use for assessment of the methods. Finally, we examine the efficiency of the proposed approaches in terms of both runtime and memory consumption.

### *4.1. Data sets*

As it is demonstrated in Table 1 we use several real-life data sets from various domains, including economy, neuroscience, epidemiology, climatology, video-surveillance, psychometric and transportation. Some statistical information about these data sets is presented in Table 2 such as tensor size, the optimum Tucker model parameter and its corresponding fit, the number of iterations for Tucker model, the percentage of non-zero values in tensor, class of values ("int" indicates integer and "float-" implies positive/negative float numbers). The mean and standard deviation of the non-zero values in tensors are also presented in the two last columns. In Appendix A the detailed description of data sets are briefly presented. Note that all the used data sets are publicly available and can be accessed via Internet.

### *4.2. Evaluation framework*

Evaluation of tensor analysis methods is still a difficult challenge. When we want to compare one method versus its counterpart, normally the model fit or error rate is measured. Also in some application domains such as chemometrics, researchers usually use some field knowledge to validate the model.

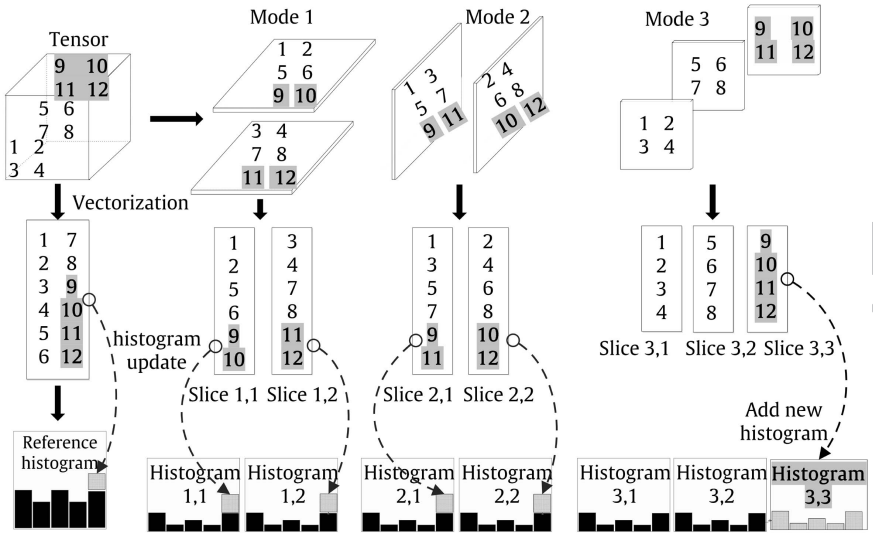



Figure 3: MASTA Update process. The initial state is shown when MASTA is already applied on tensor $\boldsymbol{X}(2 \times 2 \times 2)$ (corresponding to elements 1 to 8). At the end of this moment, MASTA outputs one reference histogram plus 6 slice histograms of (1,1),(1,2),(2,1),(2,2),(3,1),(3,2). In the next stage as is highlighted with bold font and gray background, we receive a new stream, such as S(1,1,3)=9, S(1,2,3)=10, S(2,1,3)=11, S(2,2,3)=12. We first update the reference histogram using new elements of (9, 10, 11, 12), and then update the histograms correspond to four slices (1,1),(1,2),(2,1),(2,2) that are affected by the new stream. We also generate a new histogram for slice (3,3), because all new elements belong to the third slice of the third dimension. Two histograms of slices (3,1) and (3,2) have remained unchanged, because are not affected by new elements. This example is for a time-evolving tensor (tensor that evolves only in time mode). However, MASTA allows tensor to evolve in all three directions. The only difference will be that if histogram evolves in all modes, we will not see unchanged slices like (3,1) and (3,2).

Normally, when a novel tensor methodology is developed, the computational efficiency receives more attention than accuracy. On the other side, those researchers who develop new algorithms and methods for tensor analysis do not have sufficient access to the domain knowledge for performing a realistic validation. All these issues together make the evaluation of tensor analysis a difficult challenge. In our case, this difficulty is even greater, because our proposed solution is not decomposition-based like its counterparts.

In this paper, we propose a new evaluation framework that is capable of solving this issue. This new methodology can be used for assessment of any new tensor analysis approach irrespective of the methodological details. However, we make two assumptions. The first is about the reliability of reference models where we assume:

**Assumption 3** *Tucker model with optimum model parameter (or PARAFAC with the best number of components) is the best possible model we can generate from the tensor.*

The second assumption lies within the main hypothesis of spectral-based anomaly detection techniques mentioned in [46][p. 37]: *"data can be embedded into a lower dimensional subspace in which normal instances and anomalies appear significantly different"*. By extending this for tensor-based approaches, we infer that:

**Assumption 4** *An optimum Tucker/PARAFAC solution is an effective model for discriminating the normal and abnormal features in complex tensors.*

Above assumptions state that if we apply Tucker or PARAFAC models we will be able to identify anomalies in the most effective way. Therefore, if we consider the optimum Tucker as the reference, our approach or any other tensor technique should discriminate anomalies very close to the reference models. Based on this, it is not entirely off the mark to declare that an ideal alternative tensor analysis should be able to predict top-N percentage of anomalies as accurate as the reference model.

| tensor dataset | modes | domain | description |
|---|---|---|---|
| trade [35] | $country \times country \times time$ | economy | global trade dynamics |
| eeg [36] | $frequency \times time \times channel \times trial$ | neuroscience | brain-computer interfacing |
| newmexico [37] | $location \times time \times measurements$ | epidemiology | brain cancer in N.Mexico |
| spain02 [38] | $time \times location \times measurements$ | climatology | climate analysis over Spain |
| walk [39] | $imageframes \times time$ | video-surveillance | human motion analysis |
| flightdelay [40] | $airport \times airport \times time$ | transportation | airline on-time modeling |
| flight [41] | $country \times country \times time$ | transportation | air passengers demand modeling |
| bikeboston [42] | $station \times station \times time$ | transportation | bike sharing O/D flows in Boston |
| bikewashington [43] | $station \times station \times time$ | transportation | bike sharing O/D flows in W.DC. |
| taxi [44] | $time \times region \times region$ | transportation | taxi count matrix in Beijing |
| kojimagirls [45] | $sample \times condition \times measurements$ | psychometric | behavior analysis |

Table 1: Datasets

| Dataset | Tensor size | best model | Fit | Itr. | Nnzro. | Type | Mean | STD |
|---|---|---|---|---|---|---|---|---|
| trade | $207 \times 207 \times 140$ | (3 3 2) | 25.7 | 7 | 7.7 | int+ | 3434.7 | 327.6 |
| eeg | $23 \times 350 \times 7 \times 280$ | (1 2 1 2) | 53.2 | 5 | 100 | int+ | 38.8 | 43.1 |
| newmexico | $32 \times 19 \times 23$ | (1 2 2) | 25.8 | 6 | 6.8 | int+ | 0.7 | 1.3 |
| spain02 | $699 \times 1445 \times 3$ | (3 3 2) | 61.0 | 7 | 64.0 | float- | 46.0 | 36.5 |
| walk | $288 \times 384 \times 50$ | (2 2 1) | 74.4 | 5 | 100 | int+ | 49.3 | 130.7 |
| flightdelay | $152 \times 150 \times 305$ | (2 2 1) | 10.2 | 6 | 3.2 | int+ | 66.2 | 114.8 |
| flight | $77 \times 211 \times 282$ | (3 3 1) | 30.2 | 6 | 4.4 | int+ | 7150.6 | 6161.0 |
| bikeboston | $95 \times 95 \times 327$ | (3 3 2) | 21.4 | 5 | 9.5 | int+ | 1.4 | 1.8 |
| bikewashington | $157 \times 157 \times 731$ | (2 2 1) | 17.9 | 4 | 8.5 | int+ | 1.7 | 1.9 |
| taxi | $149 \times 275 \times 196$ | (1 1 1) | 87.5 | 3 | 0.2 | int+ | 18.5 | 0.2 |
| kojimagirls | $153 \times 4 \times 20$ | (2 1 2) | 86.9 | 4 | 100 | int+ | 18.7 | 21.4 |

Table 2: Descriptive statistics of data sets

#### *4.2.1. Choosing reference model*

We test two popular tensor decomposition solutions, Tucker and PARAFAC for selecting the reference model. In order to obtain the best number of components in PARAFAC we perform CONCORDIA test [47] and for Tucker we carry out a scree test (Tucktest in N-way toolbox [48]) which are popular methods in the literature for tensor model parameter tuning. Our experiments show that the fit corresponding to the optimum model for these two decomposition techniques presents almost the same quantity for all data sets. However, the Tucker optimum model performs slightly better, because it has more flexibility in terms of imbalances in the number of components in each mode. Due to this reason, in the further experiments we only focus on the Tucker model as the reference model.

#### *4.2.2. Building reference model*

The Tucker scree test operates as follows. A candidate list of model parameters is first generated by choosing a range from a minimum (e.g. 1) to a maximum (e.g. 5). Afterward, the Tucker model is built for all the possible combinations of parameters (e.g. (1 1 1), (2 1 1),.. (5 5 5) ). Next, we calculate the explained variance for each parameter and then demonstrate every obtained value for each possible parameter on a plot called scree plot. By looking at this plot we can choose the best model parameter. A good parameter is the one that is simpler and explains more variance. For instance, between parameter (3 4 3) and (2 1 1) that respectively explain 75% and 73% of variance, the preference is to (2 1 1). Because, it has less complexity. Following this procedure, we manually obtain the optimum model parameters for each data set.

After selection of the optimum parameter for the Tucker model we apply Tucker decomposition with the obtained optimum parameter on each data set. The decomposition gives us a core tensor plus $d$ factor matrices for each mode. For a third order tensor ($n_1 \times n_2 \times n_3$) and Tucker model parameter ($r_1, r_2, r_3$), the factor matrices will be in dimensions

of $n_1 \times r_1$, $n_2 \times r_2$ and $n_3 \times r_3$, respectively for mode 1,2 and 3. The columns in the factor matrices denote the latent variables (or singular vectors). After decomposition, the features in each mode can be represented with these latent variables in a more compact way. For instance, if $r_1 = 2$, in mode 1 factor matrix is in $\mathbb{R}^2$ space where its x-axis and y-axis are respectively first and second column of the factor matrix.

Although observing anomalies in this two-dimensional space might seem straightforward, in a higher dimensional space (if $r >> 2$) it would not be so easy. In order to detect anomalies from higher dimensional spaces we require a multivariate outlier detection technique. We consider four different methods, including Wilks's method [49], Hotelling's $T^2$ [50, p. 21], Minimum Covariance Determinant (MCD) [51] and Minimum Volume Enclosing Ellipsoid (MVE) [52]. The preliminary assessment of outliers in accordance with some available prior knowledge about some data sets shows that Hotelling's $T^2$ provides more reasonable results in comparison to other methods. Therefore, we use that in further steps. Hotelling's $T^2$ statistics is computed as follows.

$$T_i^2 = (X_i - \mu)^T S^{-1} (X_i - \mu) \tag{1}$$

Where $\mu$ is the mean and $X_i$ is the multivariate observation for feature $i$, and S is the covariance matrix. We compute the $T_i^2$ for all factor matrices derived from Tucker decomposition. For instance, in the above example, we compute $T_i^2$ for factor matrices of three modes, $n_1 \times r_1$, $n_2 \times r_2$ and $n_3 \times r_3$.

The output will be $T_i^2$ for feature $i$ in each mode of tensor. Next, in order to identify the anomalous features we assume that $T^2$ follows the $\chi^2$ distribution with $k$ degrees of freedom [50, p. 23], where $k$ denotes number of columns in the factor matrices (e.g. $k = r_1$ for mode 1). Hence, features that have a significant deviation from this distribution are considered anomalies. The Cumulative Distribution Function (CDF) for the $\chi^2$ distribution [53, p. 333], which is shown in the following equation computes this deviation. Consequently, the Eq. 3 is equivalent to the statistical significance (p-value) for each feature, i.e. null hypothesis: the feature is normal.

$$CDF_i = F(T_i^2|k) = \int_0^{T_i^2} \frac{t^{(k-2)/2} e^{-t/2}}{2^{k/2}\Gamma(k/2)} dt \tag{2}$$

$$P_i = 1 - CDF_i \tag{3}$$

A lower $P$ value means a more anomalous feature. Therefore, we sort the features based on the obtained P and retrieve the top 5 percent anomalous features in each mode. We call this set "**top-5% reference anomalies**".

#### *4.2.3. Compared methods*

We use three variants of ITA framework [13] for comparison: Dynamic Tensor Analysis (DTA), Streaming Tensor Analysis (STA) and Window-based Tensor Analysis (WTA). DTA incrementally decomposes the tensor by maintaining only the covariance matrix for each arriving tensor. Then, via diagonalization it outputs the principal eigenvectors of the updated covariance matrix as projection matrices. STA attempts to approximate DTA. It instead of maintaining a covariance matrix for all arriving tensors, directly updates the principal eigenvectors using SPIRIT algorithm [54] which does not require diagonalization. STA runtime can be faster by decreasing the sampling percentage. For instance, STA with sampling percentage of 10% is almost 10 times faster than STA with 100% sampling percentage. The other algorithm, WTA instead of processing individual tensors uses a sliding window strategy for handling time dependency between consecutive tensors. It decomposes the sliding window with a regular Tucker or PARAFAC and then as well as DTA and STA keeps some statistics from the window in the processing of the next window.

The temporal information loss of ITA algorithms is not evaluated in the original papers [13, 11, 12]. So, in parallel with the evaluation of the main proposal, we evaluate this issue as well.

#### *4.2.4. Settings*

For MASTA and OHTA the input parameters are b1 and b2 which are chosen respectively 50 and 10. The required model parameters of DTA, STA and WTA are chosen equal to what is obtained for the optimum number of components in CONCORDIA test corresponding to PARAFAC. The configuration we use for WTA is the independent-window mode (IW) with Tucker decomposition. Also, no forgetting factor is used for DTA and STA. For STA we test three different sampling rates of 100% (no sampling), 50% and 10% and for WTA we examine three window sizes of 4,7 and 10.

For a fair comparison, we proceed a blind evaluation strategy. In other words, we assume that we do not know the optimum parameters for OHTA and MASTA in advance. For this reason we do not use the best b1 and b2 we obtain in Fig. 4.

#### *4.2.5. Softwares*

We use MATLAB for the experiments along with two other toolboxes, tensor toolbox [55] for computing Tucker and PARAFAC and n-way toolbox [48] for CONCORDIA and Tucker scree plot (tucktest). MATLAB implementation of ITA [56] is also used for experimenting with DTA, STA and WTA. Furthermore, we use Gohistogram [57], the Golang implementation of online histogram approximation [34].

#### *4.2.6. Evaluation process*

After we generate the reference model and top 5% reference anomalies we start to identity the top 5% anomalies via MASTA, OHTA and DTA, STA and WTA. In terms of MASTA and OHTA we apply them on the tensor data and then retrieve the top 5% low support items in each mode. Concerning DTA, STA and WTA we proceed the same procedure as was explained for Tucker in section 4.2.2, i.e. applying Hotelling's $T^2$ on the non-evolving factor matrices. However, these algorithms operate incrementally on the evolving dimension of tensor which can be time or sample mode. For identification of anomalies in the evolving mode, we perform the same process that was mentioned in [13]. In terms of DTA and STA we incrementally measure reconstruction error (residual) and then retrieve the top 5% highest errors. Regarding WTA we proceed by the similar strategy with this difference, that the reconstruction error is considered for the starting item in the window.

Once we retrieve the top 5% anomalies in each dimension using our methods and ITA algorithms we compute the accuracy of them in prediction of the *top-5 % reference anomalies* obtained by the Tucker optimum model. We define accuracy as the number of true predictions divided by the total number of reference anomalies. However, if we rely on the only single point accuracy, our assessment would be highly dependent on the anomaly border point we choose. Therefore, we opt to use the average accuracy. It means that if M is the 5% of the size of the mode, we compute the accuracy for Top-M, Top-(M-1), .., and Top-1 and then compute the mean of them. Suppose that by applying one method, Top-1, Top-2 and Top-3 anomalies be detected with accuracy of 0, 0.5 and 0.66. If we rely only on Top-1, we come up with this conclusion that the detector fails. Even if we lean to Top-3 we may conclude that the detector has succeeded, while neither of them are indeed fair. The average accuracy in this case is (0+0.5+0.66)/3=0.39. As we see, the averaging strategy fines the detector for not detecting the Top-1 but not much sever as the point-based evaluation. Moreover, it does not reward the detector for high accuracy of Top-3, because has not been such good for Top-1 and Top-2.

## 5. Result and discussion

In this section we report the result of the evaluation process explained in the previous section and then discuss the results.

Throughout this section when we use the term accuracy we refer the average accuracy defined in the previous section. Table 3 demonstrates the accuracy of each method in detecting the top 5% reference anomalies for each data set. ITA algorithms increment over the time mode, therefore for a fair comparison we separate the evolving mode from non-evolving modes. For this reason, the first column of Table 3 represent the average accuracy for evolving mode (i.e. time or sample modes) and the second column represent the mean of average accuracy for other non-evolving modes.

### *5.1. Correctness of the evaluation methodology*

According to [13, 11, 12] we expect that DTA has the better accuracy than its approximation counterpart, STA. Moreover, we expect that STA behaves similarly as DTA with no sampling percentage, but loses accuracy when sampling percentage increases. All these anticipations are satisfied and re-confirmed via obtained results in Table 3 as well. This is an evidence that our evaluation framework has been effective in assessment of tensor analysis techniques.

### *5.2. Evaluation on non-evolving modes*

In the following subsection we discuss the results corresponding to non-evolving modes (”nt” in Table 3).

| Data set | MASTA | | OHTA | | DTA | | STA | | | | | | WTA | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | s=100% | | s=50% | | s=10% | | w=4 | | w=7 | | w=10 | |
| | t | nt | t | nt | t | nt | t | nt | t | nt | t | nt | t | nt | t | nt | t | nt |
| trade | 60 | 34 | 37 | 61 | 0 | 72 | 0 | 70 | 0 | 70 | 0 | 71 | 6 | 2 | 0 | 9 | 0 | 9 |
| eeg | 40 | 15 | 31 | 14 | 17 | 51 | 17 | 46 | 14 | 45 | 14 | 28 | 24 | 47 | 18 | 44 | 18 | 43 |
| newmexico | 0 | 50 | 0 | 50 | 0 | 100 | 0 | 50 | 0 | 50 | 0 | 13 | 0 | 0 | 0 | 0 | 0 | 13 |
| spain02 | 12 | 67 | 48 | 72 | 0 | 46 | 0 | 96 | 0 | 45 | 0 | 95 | 0 | 65 | 0 | 65 | 1 | 69 |
| walk | 0 | 25 | 0 | 20 | 28 | 92 | 28 | 34 | 28 | 21 | 28 | 13 | 0 | 5 | 0 | 6 | 0 | 6 |
| flightdelay | 81 | 64 | 81 | 59 | 0 | 85 | 0 | 83 | 0 | 79 | 0 | 52 | 0 | 13 | 0 | 1 | 0 | 16 |
| flight | 0 | 36 | 0 | 33 | 28 | 87 | 31 | 73 | 31 | 65 | 23 | 50 | 0 | 0 | 0 | 0 | 0 | 2 |
| bikeboston | 11 | 64 | 6 | 64 | 12 | 84 | 13 | 82 | 13 | 84 | 13 | 89 | 4 | 0 | 1 | 0 | 0 | 0 |
| bikewashington | 23 | 60 | 0 | 59 | 56 | 99 | 57 | 97 | 49 | 92 | 18 | 39 | 9 | 1 | 5 | 18 | 10 | 2 |
| taxi | 0 | 41 | 0 | 43 | 48 | 62 | 13 | 56 | 13 | 56 | 13 | 60 | 17 | 62 | 0 | 63 | 0 | 62 |
| kojimagirls | 2 | 50 | 36 | 100 | 0 | 100 | 0 | 50 | 0 | 100 | 0 | 50 | 21 | 50 | 21 | 50 | 21 | 50 |

Table 3: Average accuracy in predication of top-5% reference anomalies. The columns identified with "t" correspond to the temporal (or sample) mode and columns with "nt" represent the average accuracy for non-temporal modes.

*5.2.1. ITA algorithms*

From Table 3 we can observe that the best accuracy is obtained via DTA. Out of 11 data sets, DTA presents accuracy over 50% in 10 data sets and in average 80% accuracy for all data sets. STA with 100% sampling rate also presents accuracy of greater than 50% in 9 data sets. By increasing the sampling rate to 50% we do not see a considerable accuracy loss comparing the 100% sampling rate. However, increasing sampling rate to 10% results in STA failure in two further data sets (newmexico and bikewashington).

WTA is not compared against DTA and STA in original paper [13], so Table 3 can be considered the first official comparison of these approaches. As we observe, WTA irrespective of window size except three data sets (spain02, taxi, kojimagirls) fails in the rest of data sets. Even increasing or decreasing the window size does not affect the performance significantly. In some data sets, accuracy gets better by increasing the window size and in some declines. However, the low performance of WTA probably relates to the fact that WTA processes tensor in higher scales and assumes that data does not contain important fluctuations inside the window. Therefore, an anomaly that occurs in one day appears normal in a 10-day window. One can infer that WTA does not have enough sensitivity to small fluctuation occurrences in the original temporal scale.

*5.2.2. MASTA*

MASTA, has been able to present an accuracy of more than 50% in six data sets which is a good result for a full approximation technique. Matching Table 3 with data sets characteristics in Table 2 shows that MASTA is vulnerable in dealing with three kinds of tensors: very dense, very sparse and high-variation tensors. Among the five data sets that MASTA has a low performance on, two sets, i.e. eeg and walk are the two most dense tensors (100% dense) and one (taxi) is the most sparse tensor (99.8% sparse). Two other data sets, flight and trade are also the top-2 in terms of the standard deviation magnitude. The best performance of MASTA is seen on bike data sets (bikeboston and bikewashington) and spain02 which have reasonable sparsity and variance. The good performance of MASTA on the two bike data sets reveals that MASTA is robust within the domain.

MASTA is weak in facing with very dense tensors because of two reasons, first it is not based on the correlation concept, therefore it loses the existing strong spatial correlations in dense tensors. It also is not an ideal technique for very sparse tensors, because, histograms for these tensors are not informative enough to be used for matching the slices, resulting in misleading judgments. MASTA is also unable to approximate the tensor with high amount of variance, because of the inherent limitation of streaming histogram calculation. As is mentioned in [34] when data distribution is highly skewed, the accuracy of histogram approximation technique decays. In high variance data sets, such property appears in its extreme form. This is the reason why we see that the offline approach (OHTA) outperforms MASTA up to 30% in terms of trade data set. However, the same justification does not hold for the flight data set. Probably it is because flight data set has a mixture of two factors. Out of 11 data sets, flight data set is the

second sparser tensor with the biggest standard deviation. It seems that OHTA has the same vulnerability as MASTA when both of these factors get involved.

#### 5.2.3. Online vs. offline histogram calculation

The results in Table 3 reveal that OHTA also has a similar performance as MASTA with slight differences in some datasets. The severe difference of these approaches is in some data sets such as trade which can be explained by the weakness of online histogram technique in handling tensors with large dispersion. This reveals that the online histogram strategy has been quite effective, so we lose only 5-6% average accuracy, in exchange of huge efficiency.

#### 5.2.4. Effect of approximation on the accuracy

To study the effect of approximation level on the accuracy, we can compare DTA against STA with three different sampling rates. STA is an approach for faster approximation of DTA. Therefore, we can study this effect if we increase the level of approximation in STA. Considering the Table 3 along with Table 2 shows that dense data sets such as eeg and walk are very sensitive to approximation. The accuracy for these data sets decays by increasing the level of approximation. For instance, in walk data set, while DTA has accuracy of 92%, the performance of STA which is an approximation for DTA sharply decays to 34% ,21% and 13% respectively for sampling rates of 100%, 50% and 10%. Our histogram-based methods which in principle are approximation solutions also present a very low performance for these data sets. Therefore, we can infer that very dense tensors are more vulnerable than the other tensors against approximation solutions.

### 5.3. Evaluation of the evolving mode

Concerning the temporal mode (as is demonstrated in Table 3 with "t") we observe that all methods have difficulty in dealing with fluctuations in the time mode. The reason why accuracy for evolving mode is lower than non-evolving mode for all methods is that time or sample mode is the main cause of tensor fluctuations in the evolving data sets. Temporal analysis with lack of knowledge about the system's past behavior results in information loss. This is an important issue that requires to be carefully taken into account when we exploit incremental approaches. Incremental approaches focus more on accuracy of non-temporal mode and sacrifice the information of time mode.

Among ITA algorithms, WTA is one of the approaches that sacrifices more time information than others. It processes a group of tensors in multiple time points together, which results in less sensitivity to low-scale variations, and subsequently much more accuracy loss.

It seems that histogram-based approaches even though do not provide ideal solution for time mode, perform slightly better than ITA algorithms. For instance, MASTA in two of data sets including flightdelay and trade provides respectively high accuracy of 81% and 60% while all ITA algorithms have zero accuracy. This reveals that histograms are better tools for keeping the summarized information in time mode than the model residuals in ITA methods.

### 5.4. Sensitivity Analysis

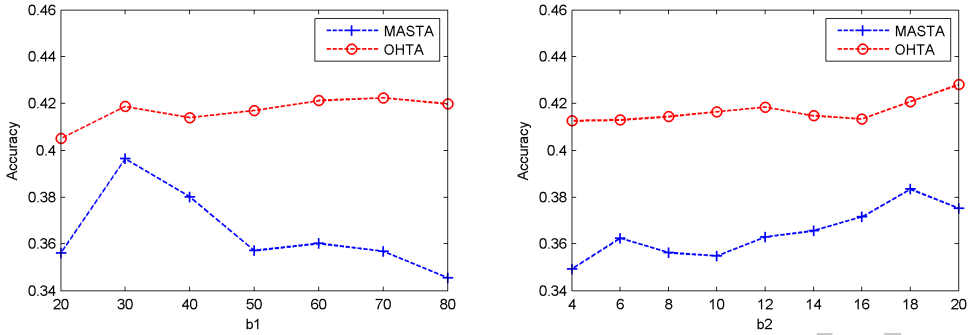
MASTA and OHTA both require two parameters b1 and b2 which are respectively the number of bins in reference/slice histograms and distance histogram. In this section we study the sensitivity of them to these parameters. To do so, we select a range of b1 from 20 to 80 and b2 from 4 to 20; apply MASTA and OHTA with varying parameters; and then compute the mean accuracy over all modes averaged for all data sets.

In order to evaluate the sensitivity, we perform a standard ANOVA test [58] on the obtained averaged accuracies. Out of 34 tests (Ten 3D tensor plus one 4D tensor) and choosing $\alpha = 0.05$, the null hypothesis "*accuracies drawn from the same distribution*" is rejected, respectively in 12 and 8 of the tests for MASTA and OHTA indicating that both methods have a moderate sensitivity to b1 parameter.

Regarding the b2 parameter, out of 34 tests only in one case, the null hypothesis is rejected for both MASTA and OHTA. This reveals that neither of both methods are sensitive to b2 parameter for a range of 4 to 20.

The sensitivity test results presented here are valid only for anomaly detection application. In some other application such as clustering, the parameter $b2$ is equivalent to the number of clusters and the proper determination of that is quite important in the model output. Sensitivity evaluation of this parameter does not make sense in such applications.

Nevertheless, for a specific application of anomaly detection, Figure 4 might be helpful while choosing the optimum parameters. We can observe that OHTA is more robust to the input parameters and its performance remains




(a) b1 (b) b2

Figure 4: Sensitivity of MASTA and OHTA to the input parameters

relatively constant by changing the parameters. However, fluctuations of MASTA show that perhaps parameters b1=30 and b2=18 lead to a slightly better detection accuracy.

### 5.5. Computational complexity

Table 4 summarizes the complexity of methods for a dense cubic tensor of $n \times n \times n$. For more simplicity, we demonstrate the dominant cost for all methods and omit all the constants such as input parameters. In the following we present the computational complexities of the proposed methods.

#### 5.5.1. OHTA

For a $n \times n \times n$ tensor the main cost for OHTA relates to the construction of the reference histogram which exploits the sorting algorithm. If we use for instance quick-sort algorithm, OHTA would require $(n^3)^2 = n^6$ time and $n^3$ space for computing the reference histogram. The construction of slice histograms also costs $3n(n^2)^2 = 3n^5$ time and $n^2$ space with additional time of $3n(b1^2)$ for EMD distance calculations. Moreover, the calculation of distance histogram $H_i$ for three modes consumes $3n^2$ time and $3n$ space. Thus, the total time complexity of OHTA is $n^6 + 3n^5 + 3n^2 + 3n(b1^2)$. Besides, OHTA requires keeping the intermediate data in the memory which adds additional cost of $n^3$ to the space complexity. Hence, the total space complexity is equal to $2n^3 + n^2 + 3n$.

#### 5.5.2. MASTA

MASTA uses the streaming histogram approximation, therefore, for a full tensor $n \times n \times n$ requires $O(n^3)$ time and $O(1)$ space for reference histogram construction. Three more $O(n^3)$ time is required for construction of the slice histograms in each mode, which makes the total time complexity $O(4n^3)$. For keeping the reference and slices histogram we need a space of $(b1)(3n + 1)$. We also do not need any space for keeping intermediate data, because every piece of tensor stream can immediately enter to MASTA for updating histograms and then get removed from the memory. Therefore, no space cost is imposed for keeping intermediate data. Hence, total space complexity of MASTA is what we need for keeping histograms which after removing the constants becomes $O(n)$.

For updating the MASTA model we need $s$ (data chunk size) time for the reference histogram and $3s$ (3 modes) for updating or adding slice histograms. Therefore, the MASTA update costs $O(4s)$. Thus, MASTA update procedure is not dependent of $n$ and is linear with data chunk size. This is an enormous advantage over ITA algorithms which at least require $O(n^2)$ for updating the model.

One of the interesting properties of MASTA is that the process of model update is separated from the model output procedure. If the user asks for model output, the input of the updated model feeds into Algorithm 3 for the final output. This process needs $3nb1^2$ time for slice reconstruction in Algorithm 4 and an additional $3nb1^2$ for histogram distance calculation. Finally, distance histograms need to be updated, that costs further $3n$. Therefore, after removing the constants, the time complexity of output procedure becomes $O(n)$. This $O(n)$ computation is not mandatory for the learning part. We only need to perform this procedure when the user asks for the model output.

*5.6. Empirical efficiency*

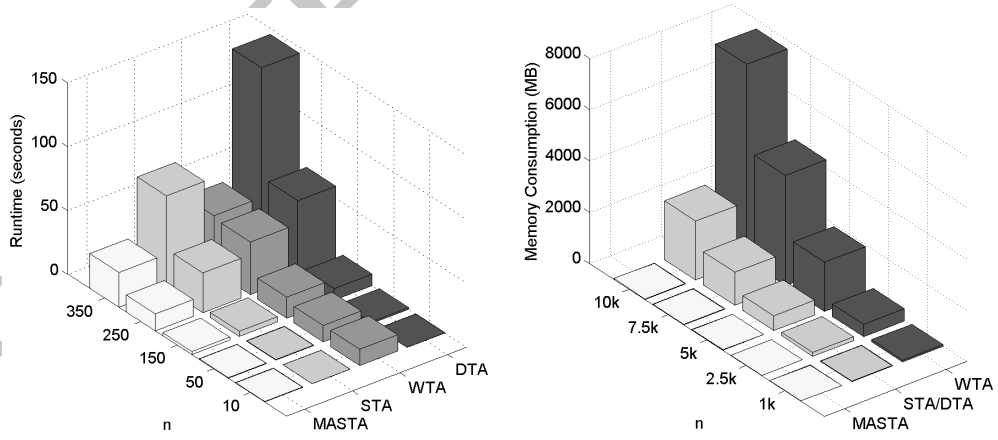
The empirical assessment for runtime is executed on a PC with Intel Core 2 Duo, 2000 MHz. The memory consumption test is also performed on a computer with 8GB of memory. We pick 50 and 5 respectively, for b1 and b2 parameters in OHTA and MASTA. We also choose 5 for the number of components in PARAFAC. For Tucker and WTA the model parameter is chosen (5 5 5) and for DTA and STA the parameters are set as (5 5). The window size of 10 is also selected for WTA.

For assessment of runtime, we create five random cubic dense tensors $n \times n \times n$, $n \in \{10, 50, 150, 250, 350\}$ and compute the runtime required for generating the model. The result is presented in Fig. 5-a. As we can see, MASTA performs slightly faster than all ITA variants.

In order to evaluate the memory consumption, we create random cubic dense tensors $n \times n \times n$, $n \in 1000 \times \{1, 2.5, 5, 7.5, 10\}$ and measure the required memory for model generation by four methods including MASTA, STA/ DTA and WTA. Fig. 5-b shows the result. As we can see, for processing the large dense tensor of $10k \times 10k \times 10k$, MASTA requires only 24 MB of memory that is a tremendous improvement over DTA/STA and WTA that respectively use 2.24 GB and 7.46 GB. Roughly speaking, we can say that for three-order cubic tensors with 100% density, MASTA can solve 100x bigger problems comparing ITA approaches.

| Type | Method | Time | Update time | Space | Estimation for n=$10^4$ |
|---|---|---|---|---|---|
| Offline | Tucker/PARAFAC | $n^4$ | — | $n^3$ | 7453 GB |
| ITA | DTA | $n^4$ | $n^3$ | $n^2$ | 2.24 GB |
| | STA | $n^3$ | $n^2$ | $n^2$ | 2.24 GB |
| | WTA | $n^3$ | $n^2$ | $n^2$ | 7.46 GB |
| Histogram-based | OHTA | $n^6$ | — | $n^3$ | 7442 GB |
| | MASTA | $n^3$ | $s$ | $n$ | 24 MB |

Table 4: Time and space complexity of tensor analysis approaches for third-order tensor of $\boldsymbol{X}(n \times n \times n)$ with zero sparsity. All constants are omitted for simplicity. $s$ denotes the size of the recent data chunk. The time complexity of Tucker/PARAFAC is given with this assumption that initialization is performed by HOSVD.




(a) Runtime (b) Memory consumption

Figure 5: Efficiency comparison of the proposed multi-aspect-streaming method (MASTA) against streaming methods for processing of three-order cubic $n \times n \times n$ tensor.

## 6. Case studies

Since we do not have access to the domain specialists, we are not able to construe the results and conform them to the reality. Any kind of interpretation without having access to a precious domain knowledge may result in a

misleading conclusion. However, we cautiously consider two event detection case studies related to two data sets: trade and walk. These two case studies correspond to one success and one failure case of MASTA in handling evolving mode. As is specified in the column "t" of Table 3, MASTA provides 60% accuracy for trade set (success) and 0% accuracy for walk data set (failure). The reason why we choose these two data sets is that there exists an evident knowledge about these data sets that can be incorporated for interpretation of results. In the former case we have the knowledge of global financial crisis that happened in 2007-2009 and for the latter we have the visual insight from the video.

### *6.1. Event detection in trade data set*

The trade data set contains historical pairwise trade volumes between 207 countries over 140 years. In this case study, we want to see how methods detect the global financial crisis happened in 2007-2009 [59]. Our expectation is that tensor analysis models rank these years as the most anomalous years. So, in this case study, we apply all methods, including Tucker and PARAFAC reference models on the trade tensor data to see how they rank the crisis period of 2007-2009 among 140 years. Here we use the same parameters used in section 4.2.4. For STA we test for the sampling rate of 50% and for WTA we examine the window size of 10.

| Year | Tucker | PARAFAC | MASTA | OHTA | DTA | STA(s=50%) | WTA(w=10) |
|---|---|---|---|---|---|---|---|
| 2007 | 4 | 4 | 2 | 2 | 23 | 73 | 131 |
| 2008 | 1 | 2 | 3 | 4 | 19 | 66 | 132 |
| 2009 | 2 | 6 | 4 | 13 | 18 | 67 | 133 |

Table 5: The anomalous ranking of the period related to global financial crisis 2007-2009 in trade data set.

The obtained rankings are presented in Table 5. As we can see, Tucker and MASTA rank the crisis years in their top-4 priority. PARAFAC with a little difference ranks the crisis years in its top-6. OHTA ranks the crisis years in the top-13 with similar orders as MASTA. Also, DTA, STA and WTA identify these years, respectively in their top-23, top-67 and top-133.

The rankings we obtain in this study have a meaningful correlation with the reported accuracies in Table 3. For instance, from Table 3 we anticipate that MASTA and OHTA present a better accuracy for evolving mode (respectively 60% and 37%) in comparison with other methods. Here, similarly we observe that rankings of MASTA and OHTA are close to the reference models. In particular, between MASTA and OHTA, the former one ranks the crisis period better than the latter which makes sense according to its finer accuracy. This can be interpreted as an evidence for validity of reported accuracies in Table 3.

### *6.2. Event detection in video data set*

Here we analyze the walk data set which includes the appearance of a human object entering the scene while walking. The background of the video is not static, so that includes some small-scale movements in the back.

We transform the color image frames to grayscale and build a three-order tensor of $x \times y \times frame$ (*or time*). Then we apply MASTA and OHTA with b1=50 and b2=3 on the video tensor to see how they detect the entrance of the human object. The three important moments in the video and the output of MASTA and OHTA on the evolving mode are illustrated in Fig. 6. As we can see, OHTA discriminates the frames very close to the reality happens in the video. It clusters the instants into three relevant moments: a) frames 1-24 that object still has not entered into the scene; b) frames 25-29 which are related to when the object is entering but not still appeared fully in the scene; and c) frames 30 to 50 that correspond to moments when the object has fully entered but still moving in the scene.

MASTA, although being able to detect the entrance event at t=25, is too sensitive to the small-scale changes. In certain moments before the object enters the scene, some small-scale changes are seen in the background. MASTA assumes that these small movements belong to a new emerging event, so it allocates a new bin for them (t=13 and t=15).

The other difficulty of MASTA relates to, when a object turns around in different directions while he enters into scene. MASTA faces with this doubt that maybe that movements belong to a new concept, so it creates a new bin for justification of the new movements (e.g. t=27 or t=34-36). However, it is unsure how to classify the object

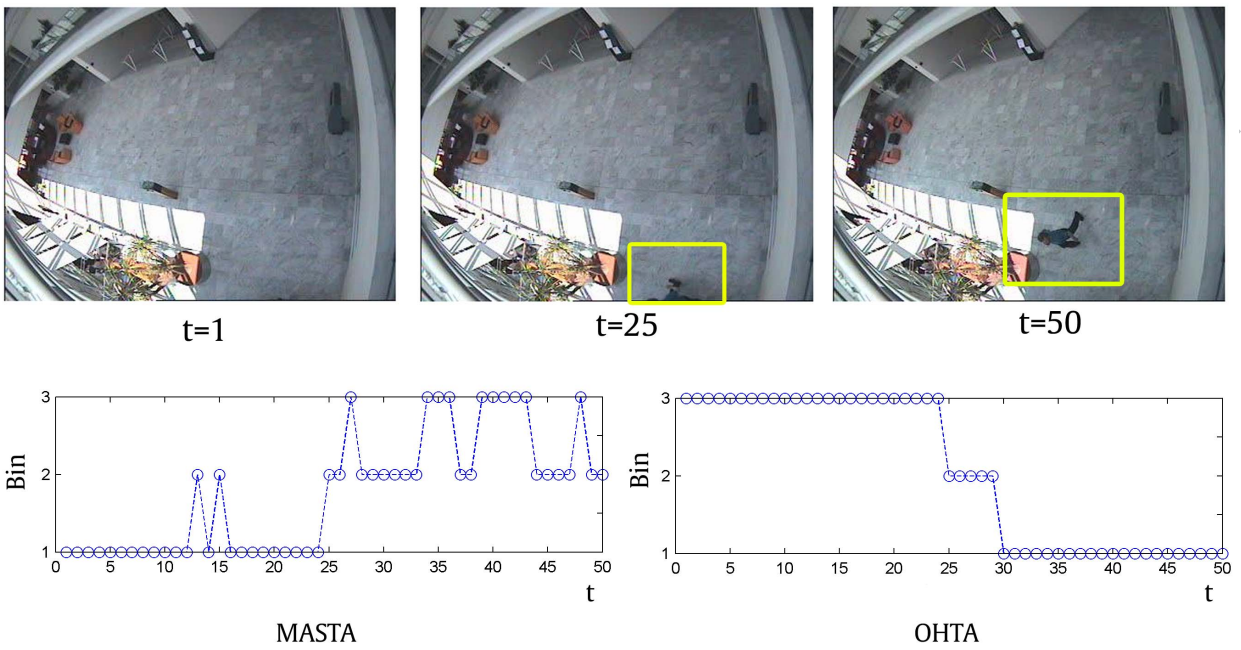



Figure 6: Output of MASTA and OHTA on the time mode (3rd mode) of walk tensor $x \times y \times time$. t=1) video record started. t=2) human object starts to enter to scene while he is walking and moving in different directions. t=3) position of object at the end of record.

movements, thus between t=25 to t=50 repeats its mistake several times. The reason of such behavior is that MASTA does not have free access to the historical data, as opposed to its offline counterpart. Hence, it is natural for it to react faster to the small-scale changes.

One interesting point is that either MASTA or OHTA present meaningful outputs which were not expected, according to their zero average accuracy in Table 3. The reason is that our evaluation framework in this study is based on quality of anomaly detection and not quality of clusters. Therefore, MASTA and OHTA even though might have less value for anomaly detection on dense tensors, might have potential in applications such as clustering and change detection. Evaluation of MASTA in these applications requires future research and new evaluation methodologies which was out of the scope of this work.

## 7. Conclusion

We study the application of histograms to tensor analysis. We introduce two histogram-based algorithms, namely OHTA and MASTA respectively, for offline and streaming analysis. The streaming solution not only presents a close accuracy to the offline approach, but also reduces the total time complexity from $O(n^6)$ to $O(n^3)$ and space complexity from $O(n^3)$ to $O(n)$. More importantly, it allows tensor evolution through all modes, which is a major progress. It also has three superiorities over the state-of-the-art incremental tensor analysis techniques: Firstly, MASTA has a significant lower time complexity for updating the model when new data arrives: $O(s)$ vs. $O(n^2)$; Secondly, it is robust against the well-known problem of *intermediate data explosion*; and finally, it consumes much less space: $O(n)$ vs $O(n^2)$, such that is capable to solve bigger problems.

MASTA, however, suffers from two main issues: First, is not a tensor decomposition approach, hence its application is limited to the analysis-only tasks such as anomaly detection, clustering or change detection; Second, its accuracy is not ideal as other approximation solution such as DTA. Its best accuracy is 67% for non-evolving mode and 81% for evolving mode. As for the last, it has a poor performance in handling some kind of tensors including very dense, very sparse and tensors with large amount of dispersion.

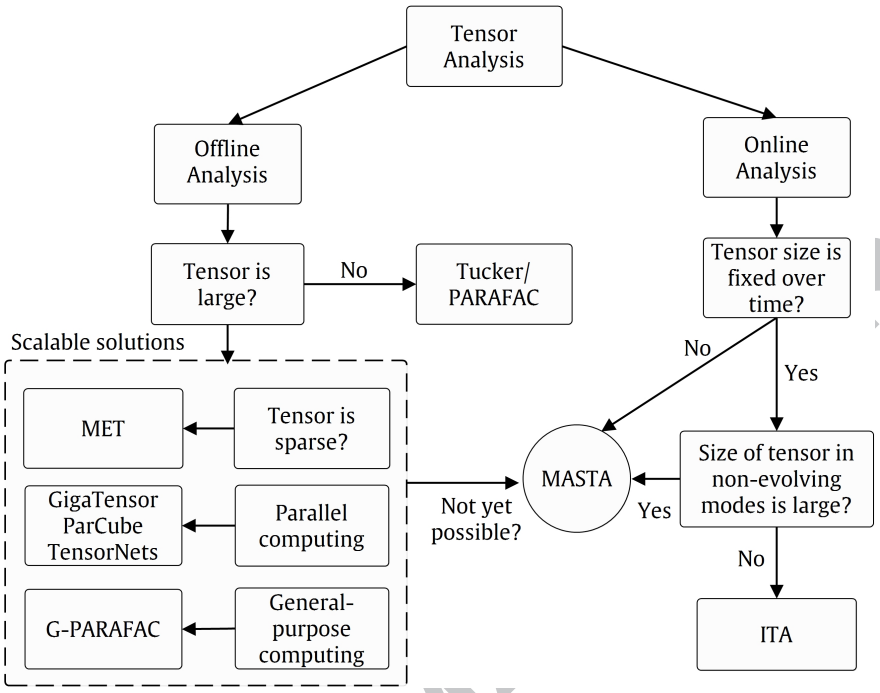



Figure 7: A guideline for choosing the appropriate tensor analysis solution

We recommend MASTA as the last alternative solution for three situations (see Fig. 7): 1) when scalable tensor decomposition solutions are not applicable due to the shortages in equipments and infrastructures; 2) when tensor size changes over time and the model needs to be updated rapidly and frequently; and 3) when size of tensor is large and is not affordable by ITA. In such cases, MASTA can offer a better solution at least finer than WTA or comparable to STA with 5-10% sampling rate.

Future works include evaluation of MASTA on other problems such as clustering and concept drift detection; investigation of MASTA problems in particular settings such as very dense or very sparse tensors; and development of a parallel version of MASTA.

## Acknowledgment

This research was supported by the Projects NORTE-07-0124-FEDER-000059/000056 which is financed by the North Portugal Regional Operational Program (ON.2 O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF), and by national funds, through the Portuguese funding agency, Fundaٰo para a Ciłncia e a Tecnologia (FCT). The authors also acknowledge the support of the European Commission through the project MAESTRA (Grant Number ICT-750 2013-612944). The authors thank AEMET and UC for the data provided for this work (Spain02 dataset, `http://www.meteo.unican.es/datasets/spain02`). The authors thank the anonymous reviewers for their constructive reviews.

## Appendix A. Data sets

*trade.* This dataset [35] includes the bilateral trade flows (import/export in US dollars) between countries for the period from 1870-2009. We transform the raw data to tensor scheme of $country \times country \times year$.

*eeg.* BCI III motor imagery dataset (4a) data set is made as a part of brain-computer interface study in [36]. According to the instructions in [60] we extract trials from EEG continuous signals in this dataset to a four-dimensional tensor of $features \times time \times channel \times trail$.

*newmexico.* Brain cancer incidence in New Mexico data set [37] includes the New Mexico Tumor Registry between the years of 1973 to 1991 for 32 sub-regions of the New Mexico state along with some categorical demographic features that together form 23 features. We generate a three-way tensor with the county in the first mode, year in the second mode and measurements in the third mode.

*spain02.* Spain02 data set [38] contains the 50-year high-resolution monthly gridded precipitation data over Spain. First and second modes of tensor for this data sets are respectively month and the spatial grid-id and third mode contains precipitation and (maximum and minimum) temperature.

*walk.* This video data set [39] is extracted from CAVIAR project data sets which is filmed at INRIA Labs at Grenoble, France. The resolution of videos is in half-resolution PAL standard (384 x 288 pixels, 25 frames per second). A number of video clips were recorded in this project acting out the different scenarios of interest. In this work, we use the first 50 frames of CAVIAR /INRIA walking data set (walk1) where one human object appears in the video after a while of recording.

*flightdelay.* This data set is a part of Data expo 09 set (airline on-time performance competition) [40] consists of flight arrival and departure delays for all commercial flights within the USA, in 2008. The tensor is in scheme of $origin \times destination \times time$ and elements in the tensor are the average daily delays measured for corresponding flights.

*flight.* USA international air passenger statistics [41] that reports the commercial traffic traveling between international points and U.S. airports from 1990 to 2013. We transform the raw data to a flow tensor $airport \times airport \times month$ in this study.

*bikeboston.* This dataset has been extracted from hub-way data challenge 2013 [42]. It includes a historical usage log of all transactions in the Boston bike sharing system from 2011-07-28 to 2012-10-01, exclusive of the system's off-days in winter, a total of 327 days.

*bikewashington.* . Washington, D.C. bike-sharing dataset [43] includes a historical usage log of all transactions in the bike-sharing network in a two-year window from 2011-01-01 to 2012-12-31, in total, 731 days. We select top 157 stations that have more frequent trips.

*taxi.* This data set [44] contains a one-week trajectories of 10,357 taxis in Beijing, China, in a period between 2008-02-02 to 2008-02-08. We use a grid strategy to divide the spatial space into equal areas. Then in each zone, we calculate the number of existing taxis in each hour (for total 149 hours).

*kojimagirls.* This data set [45] includes the judgments (in 20 scales) of 153 parents behavior with respect to their own 13-year child on four conditions of daughter-father, daughter-mother, father-father and mother-mother. The output tensor consist of a $subjects \times scales \times condition$.

## References


[1] M. Mørup, Applications of tensor (multiway array) factorizations and decompositions in data mining, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 1 (1) (2011) 24–40.

[2] L. R. Tucker, Some mathematical notes on three-mode factor analysis, Psychometrika 31 (3) (1966) 279–311.

[3] R. A. Harshman, Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis, UCLA Working Papers in Phonetics 16 (1) (1970) 84.

[4] E. Acar, D. M. Dunlavy, T. G. Kolda, M. Mørup, Scalable tensor factorizations for incomplete data, Chemometrics and Intelligent Laboratory Systems 106 (1) (2011) 41–56.

[5] U. Kang, E. E. Papalexakis, A. Harpale, C. Faloutsos, Gigatensor: scaling tensor analysis up by 100 times - algorithms and discoveries, in: The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012, 2012, pp. 316–324. doi:10.1145/2339530.2339583.
URL http://doi.acm.org/10.1145/2339530.2339583

[6] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM 51 (1) (2008) 107–113.

[7] A. L. F. de Almeida, A. Y. Kibangou, Distributed large-scale tensor decomposition, in: IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014, 2014, pp. 26–30. doi:10.1109/ICASSP.2014.6853551.
URL http://dx.doi.org/10.1109/ICASSP.2014.6853551

[8] A. L. De Almeida, A. Y. Kibangou, Distributed computation of tensor decompositions in collaborative networks, in: Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on, IEEE, 2013, pp. 232–235.

[9] E. E. Papalexakis, C. Faloutsos, N. D. Sidiropoulos, Parcube: Sparse parallelizable tensor decompositions, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2012, pp. 521–536.

[10] D. Chen, X. Li, L. Wang, S. Khan, J. Wang, K. Zeng, C. Cai, Fast and scalable multi-way analysis of neural data, Computers, IEEE Transactions on PP (99) (2014) 1–1. doi:10.1109/TC.2013.2295806.

[11] J. Sun, D. Tao, C. Faloutsos, Beyond streams and graphs: dynamic tensor analysis, in: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2006, pp. 374–383.

[12] J. Sun, S. Papadimitriou, S. Y. Philip, Window-based tensor analysis on high-dimensional and multi-aspect streams., in: ICDM, Vol. 6, 2006, pp. 1076–1080.

[13] J. Sun, D. Tao, S. Papadimitriou, P. S. Yu, C. Faloutsos, Incremental tensor analysis: Theory and applications, ACM Transactions on Knowledge Discovery from Data (TKDD) 2 (3) (2008) 11.

[14] T. G. Kolda, J. Sun, Scalable tensor decompositions for multi-aspect data mining, in: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 363–372.

[15] L. Shi, A. Gangopadhyay, V. P. Janeja, Stensr: Spatio-temporal tensor streams for anomaly detection and pattern discovery, Knowledge and Information Systems (2014) 1–21.

[16] H. Kim, S. Lee, X. Ma, C. Wang, Higher-order pca for anomaly detection in large-scale networks, in: Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2009 3rd IEEE International Workshop on, IEEE, 2009, pp. 85–88.

[17] K. Glass, R. Colbaugh, M. Planck, Automatically identifying the sources of large internet events, in: Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on, IEEE, 2010, pp. 108–113.

[18] M. A. Prada, J. Toivola, J. Kullaa, J. Hollmén, Three-way analysis of structural health monitoring data, Neurocomputing 80 (2012) 119–128.

[19] S. Lee, H. Liu, M. Kim, J. T. Kim, C. Yoo, Online monitoring and interpretation of periodic diurnal and seasonal variations of indoor air pollutants in a subway station using parallel factor analysis (parafac), Energy and Buildings 68 (2014) 87–98.

[20] H.-H. Mao, C.-J. Wu, E. E. Papalexakis, C. Faloutsos, K.-C. Lee, T.-C. Kao, Malspot: Multi2 malicious network behavior patterns analysis, in: Advances in Knowledge Discovery and Data Mining, Springer, 2014, pp. 1–14.

[21] A. Baum, A. S. Meyer, J. L. Garcia, M. Egebo, P. W. Hansen, J. D. Mikkelsen, Enzyme activity measurement via spectral evolution profiling and parafac, Analytica chimica acta 778 (2013) 1–8.

[22] S. Hemissi, I. R. Farah, K. Saheb Ettabaa, B. Solaiman, Multi-spectro-temporal analysis of hyperspectral imagery based on 3-d spectral modeling and multilinear algebra, Geoscience and Remote Sensing, IEEE Transactions on 51 (1) (2013) 199–216.

[23] T. G. Kolda, B. W. Bader, Tensor decompositions and applications, SIAM review 51 (3) (2009) 455–500.

[24] Y. Ioannidis, The history of histograms (abridged), in: Proceedings of the 29th international conference on Very large data bases-Volume 29, VLDB Endowment, 2003, pp. 19–30.

[25] J. Gama, Knowledge discovery from data streams, Chapman & Hall/CRC Boca Raton, 2010.

[26] S. Guha, N. Koudas, K. Shim, Data-streams and histograms, in: Proceedings of the thirty-third annual ACM symposium on Theory of computing, ACM, 2001, pp. 471–475.

[27] M. Datar, A. Gionis, P. Indyk, R. Motwani, Maintaining stream statistics over sliding windows, SIAM Journal on Computing 31 (6) (2002) 1794–1813.

[28] A. C. König, G. Weikum, Combining histograms and parametric curve fitting for feedback-driven query result-size estimation, in: Proceedings of the 25th International Conference on Very Large Data Bases, Morgan Kaufmann Publishers Inc., 1999, pp. 423–434.

[29] V. Poosala, Y. E. Ioannidis, Estimation of query-result distribution and its application in parallel-join load balancing, in: VLDB, Citeseer, 1996, pp. 448–459.

[30] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Content-based image retrieval at the end of the early years, Pattern Analysis and Machine Intelligence, IEEE Transactions on 22 (12) (2000) 1349–1380.

[31] G. Pass, R. Zabih, Histogram refinement for content-based image retrieval, in: Applications of Computer Vision, 1996. WACV'96., Proceedings 3rd IEEE Workshop on, IEEE, 1996, pp. 96–102.

[32] Y. Rubner, C. Tomasi, L. J. Guibas, The earth mover's distance as a metric for image retrieval, International Journal of Computer Vision 40 (2) (2000) 99–121.

[33] S. Guha, N. Koudas, K. Shim, Approximation and streaming algorithms for histogram construction problems, ACM Transactions on Database Systems (TODS) 31 (1) (2006) 396–438.

[34] Y. Ben-Haim, E. Tom-Tov, A streaming parallel decision tree algorithm, The Journal of Machine Learning Research 11 (2010) 849–872.

[35] K. Barbieri, O. M. Keshk, B. M. Pollins, Trading data evaluating our assumptions and coding rules, Conflict Management and Peace Science 26 (5) (2009) 471–491.

[36] G. Dornhege, B. Blankertz, G. Curio, K. Muller, Boosting bit rates in noninvasive eeg single-trial classifications by feature combination and multiclass paradigms, Biomedical Engineering, IEEE Transactions on 51 (6) (2004) 993–1002.

[37] M. Kulldorff, Brain cancer incidence in New Mexico, `http://www.satscan.org/datasets/nmbrain/index.html`, accessed: December 2012 (2012).

[38] S. Herrera, J. M. Gutiérrez, R. Ancell, M. Pons, M. Frías, J. Fernández, Development and analysis of a 50-year high-resolution daily gridded precipitation dataset over spain (spain02), International Journal of Climatology 32 (1) (2012) 74–85.

[39] The School of Informatics, University of Edinburgh, Clips from INRIA (1st Set), `http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/`, accessed: June 2014 (2014).

[40] ASA Section on Statistical Computing, Data expo 2009, `http://stat-computing.org/dataexpo/2009/`, accessed: June 2014 (2014).

[41] U.S. Department of Transportation, U.s. international air passenger and freight statistics report, `http://www.dot.gov/policy/aviation-policy/us-international-air-passenger-and-freight-statistics-report` (June 2013).

[42] Hubway, Hubway data visualization challenge, `http://hubwaydatachallenge.org/` (June 2013).

[43] CapitalBikeShare, Capital bikeshare trip history data, `http://capitalbikeshare.com/trip-history-data` (March 2013).

[44] J. Yuan, Y. Zheng, X. Xie, G. Sun, Driving with knowledge from the physical world, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2011, pp. 316–324.
[45] H. Kojima, Inter-battery factor analysis of parents' and children's reports of parental behavior., Japanese Psychological Research.
[46] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Computing Surveys (CSUR) 41 (3) (2009) 15.
[47] R. Bro, H. A. Kiers, A new efficient method for determining the number of components in parafac models, Journal of chemometrics 17 (5) (2003) 274–286.
[48] C. A. Andersson, R. Bro, The n-way toolbox for matlab, Chemometrics and Intelligent Laboratory Systems 52 (1) (2000) 1–4.
[49] S. S. Wilks, Multivariate statistical outliers, Sankhyā: The Indian Journal of Statistics, Series A (1963) 407–426.
[50] R. L. Mason, J. C. Young, Multivariate statistical process control with industrial applications, Vol. 9, Siam, 2002.
[51] P. J. Rousseeuw, Least median of squares regression, Journal of the American statistical association 79 (388) (1984) 871–880.
[52] P. Sun, R. M. Freund, Computation of minimum-volume covering ellipsoids, Operations Research 52 (5) (2004) 690–706.
[53] R. A. Thisted, Elements of statistical computing: numerical computation, Vol. 1, CRC Press, 1988.
[54] S. Papadimitriou, P. Yu, Optimal multi-scale patterns in time series streams, in: Proceedings of the 2006 ACM SIGMOD international conference on Management of data, ACM, 2006, pp. 647–658.
[55] B. W. Bader, T. Kolda, et al., Matlab tensor toolbox version 2.5, `http://www.sandia.gov/~tgkolda/TensorToolbox`, accessed: December 2012 (2012).
[56] J. Sun, Incremental tensor analysis, `http://www.dasfa.net/wiki/index.php?title=Jimeng_Sun`, accessed: December 2012 (2012).
[57] VividCortex, gohistogram package, `https://github.com/VividCortex/gohistogram`, accessed: September 2014 (2014).
[58] D. C. Montgomery, D. C. Montgomery, D. C. Montgomery, Design and analysis of experiments, Vol. 7, Wiley New York, 1984.
[59] V. Acharya, T. Philippon, M. Richardson, N. Roubini, The financial crisis of 2007-2009: Causes and remedies, Financial Markets, Institutions & Instruments 18 (2) (2009) 89–137.
[60] A. H. Phan, Nfea: Tensor toolbox for feature extraction and applications (2011).

- We extend the application of histograms to tensor analysis problem
- We propose the first approach for multi-aspect-streaming tensor analysis (MASTA)
- MASTA is space-efficient, fast and constant-time for update
- We evaluate the strengths and weaknesses of MASTA on 11 real-life data sets
- The proposed approach is useful for both streaming and scalable problems