Window-based Tensor Analysis on High-dimensional and Multi-aspect Streams

Jimeng Sun[§] Spiros Papadimitriou[†] Philip S. Yu[†]

[§] Carnegie Mellon University
Pittsburgh, PA, USA

[†] IBM T.J. Watson Research Center
Hawthorne, NY, USA

Abstract

Data stream values are often associated with multiple aspects. *For example, each value from environmental sensors may have an associated type (e.g., temperature, humidity, etc) as well as location. Aside from timestamp, type and location are the two additional aspects. How to model such streams? How to simultaneously find patterns within and across the multiple aspects? How to do it incrementally in a streaming fashion? In this paper, all these problems are addressed through a general data model, tensor streams, and an effective algorithmic framework, window-based tensor analysis (WTA). Two variations of WTA, independent-window tensor analysis (IW) and moving-window tensor analysis (MW), are presented and evaluated extensively on real datasets. Finally, we illustrate one important application, Multi-Aspect Correlation Analysis (MACA), which uses WTA and we demonstrate its effectiveness on an environmental monitoring application.*

1 Introduction

Data streams have received attention in different communities due to emerging applications, such as environmental monitoring and sensor networks [7]. The data are modelled as a number of co-evolving streams (time series with an increasing length). Most data mining operations need to be redesigned for data streams because of the streaming requirement, i.e., the mining result has to be updated efficiently for the newly arrived data.

In the standard stream model, each value is associated with a $(\text{timestamp}, \text{stream-id})$ pair. However, the stream-id itself may have some additional structure. For example, it may be decomposed into $(\text{location-id}, \text{type}) \equiv \text{stream-id}$. We call each such component of the stream-id an *aspect*. The number of discrete values each aspect may take is called its *dimensionality*, e.g., the type aspect has dimensionality 4 and the individual dimensions are temperature, humidity and etc. This additional structure should not be ignored in data exploration tasks, since it may provide additional insights. Thus the typical "flat-world view" may be insufficient. In summary, even though the traditional data stream model is quite general, it cannot easily capture some important aspects of the data.

In this paper, we tackle the problem at three levels. First, we address the issue of modeling such high-dimensional and multi-aspect streams. We present the *tensor stream* model, which generalizes multiple streams into high-order tensors represented using a sequence of multi-arrays.

Second, we study how to summarize the tensor stream efficiently. We generalize the moving/sliding window model from a single stream to a tensor stream. Every tensor window includes multiple tensors. Each of these tensors corresponds to the multi-aspect set of measurements associated with one timestamp. Subsequently, using multilinear analysis [3] which is a generalization of matrix analysis, we propose *window-based tensor analysis (WTA)* for tensor streams, which summarizes the tensor windows efficiently, using small core tensors associated with different projection matrices, where core tensors and projection matrices are analogous to the singular values and singular vectors for a matrix. Two variations of the algorithms for WTA are proposed: 1) *independent-window tensor analysis (IW)* which treats each tensor window independently; 2) *moving-window tensor analysis (MW)* which exploits the time dependence across neighboring windows to reduce computational cost significantly.

Third, we introduce an important application using WTA, which demonstrates its practical significance. In particular, we describe *Multi-Aspect Correlation Analysis (MACA)*, which *simultaneously* finds correlations within a single aspect and also across multiple aspects.

In summary, our main contributions are the following:

- **Data model**: We introduce tensor streams to deal with high-dimensional and multi-aspect streams.
- **Algorithmic framework**: We propose window-based tensor analysis (WTA) to effectively extract core patterns from tensor streams.
- **Application**: Based on WTA, multi-aspect correlation analysis (MACA) is presented to simultaneously compute the correlation within and across all aspects.

2 Problem Formulation

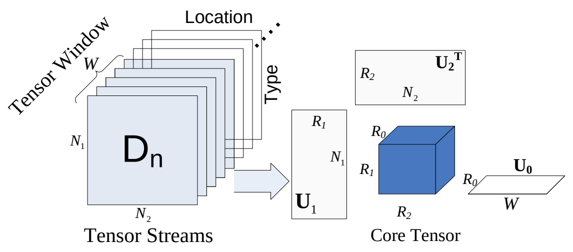


Figure 1. window-based tensor analysis

In this section, we first formally define the notions of *tensor stream* and *tensor window*. Then we formulate two versions of window-based tensor analysis (WTA). Recall that a tensor mode corresponds to an aspect of the data streams.

Definition 1 (Tensor stream) *A sequence of Mth order tensors $\mathcal{D}_1 \dots \mathcal{D}_n$, where each $\mathcal{D}_i \in \mathbb{R}^{N_1 \times \cdots \times N_M}$ ($1 \le i \le n$) and n is an integer that increases with time, is called a tensor stream denoted as $\{\mathcal{D}_i \mid 1 \le i \le n, n \to \infty\}$.*

We can consider a tensor stream is built incrementally over time. The most recent tensor in the stream is $\mathcal{D}_n$. In the environmental monitoring example, a new 2nd order tensor (as shown in Figure 1) arrives every minute.

Definition 2 (Tensor window) *A tensor window $\mathcal{D}(n, W)$ consists of a subset of a tensor stream ending at time n with size W. Formally $\mathcal{D}(n, w) \in \mathbb{R}^{W \times N_1 \times \ldots \times N_M} = \{\mathcal{D}_{n-w+1}, \dots, \mathcal{D}_n\}$ where each $\mathcal{D}_i \in \mathbb{R}^{N_1 \times \cdots \times N_M}$.*

A tensor window localizes the tensor stream into a (smaller) tensor sequence with cardinality W and at particular time n. The current tensor window refers to the window ending at the current time. Notice that the tensor window is a natural high-order generalization of the sliding window model in data streams.

The goal of *window-based tensor analysis (WTA)* is to incrementally extract patterns from high-dimensional and multi-aspect streams. In this paper, we formulate this pattern extraction process as a dimensionality reduction problem on tensor windows. Two versions of WTA are presented as follows:

Problem 1 (Independent-window tensor analysis (IW)) *Given a tensor window $\mathcal{D} \in \mathbb{R}^{W \times N_1 \times \cdots \times N_M}$, find the projection matrices $\mathbf{U}_0 \in \mathbb{R}^{W \times R_0}$ and $\mathbf{U}_i \in \mathbb{R}^{N_i \times R_i}\,|_{i=1}^{M}$ such that the reconstruction error is minimized: $e = \left\| \mathcal{D} - \mathcal{D} \prod_{i=0}^{M} \times_i (\mathbf{U}_i \mathbf{U}_i^T) \right\|_F^2$*

The core tensor $\mathcal{Y}$ is defined as $\mathcal{D} \prod_{i=0}^{M} \times_i \mathbf{U}_i$. Intuitively, a projection matrix specifies the "concepts" along a given aspect/mode, while the core tensor consists of the "concept" association across all aspects/modes. Note that for 2nd-order tensors (i.e., matrices), the core tensor is the diagonal matrix Σ of singular values.

Problem 2 (Moving-window tensor analysis (MW)) *Given the current tensor window $\mathcal{D}(n, W) \in \mathbb{R}^{W \times N_1 \times \cdots \times N_M}$ and the old result for $\mathcal{D}(n-1, W)$, find the new projection matrices $\mathbf{U}_0 \in \mathbb{R}^{W \times R_{0,n}}$ and $U_i \in \mathbb{R}^{N_i \times R_{i,n}}\,|_{i=1}^{M}$ such that the reconstruction error is minimized: $e = \left\| \mathcal{D}(n, W) - \mathcal{D}(n, W) \prod_{i=0}^{M} \times_i (\mathbf{U}_i \mathbf{U}_i^T) \right\|_F^2$*

Finally, we illustrate a mining application using the output of WTA in Section 5.

3 Window-based Tensor analysis

Section 3.1 first introduces the goal and insight of the general window-based tensor analysis, where we point out the importance of good initialization for the algorithm. Next we propose two algorithms, independent-window and moving-window tensor analysis based on different initialization strategies in Section 3.2.

3.1 Iterative Optimization on windows

The goal of tensor analysis is to find the set of orthonormal projection matrices $\mathbf{U}_i|_{i=0}^{M}$ that minimize the reconstruction error $d(\mathcal{D}, \tilde{\mathcal{D}})$, where $d(\cdot, \cdot)$ is a divergence function between two tensors; $\mathcal{D}$ is the input tensor; $\tilde{\mathcal{D}}$ is the approximation tensor defined as $\mathcal{D} \prod_{i=0}^{M} \times_i (\mathbf{U}_i \mathbf{U}_i^T)$.

The principle is to optimize parameters one at a time by fixing all the rest. The benefit comes from the simplicity and robustness of the algorithms. An iterative meta-algorithm for window-based tensor analysis is shown in Figure 2.

To instantiate the algorithm, three things are needed:
Initialization condition: This turns out to be the crucial component for data streams. Different schemes for this are presented in Section 3.2.
Optimization strategy: This is closely related to the divergence function $d(\cdot, \cdot)$. Gradient descent type of methods can be developed in most of cases. However, in this paper, we use the square Frobenius norm $\| \cdot \|_F^2$ as the divergence function, which naturally leads to a simpler and faster iterated method, alternating least squares. More specifically, line 4 of Figure 2 is replaced by the following steps:

1. Construct $\mathcal{D}^d = \mathcal{D}(\prod_{i \ne d} \times_i \mathbf{U}_i) \in \mathbb{R}^{R_0 \times \cdots \times N_d \times \cdots \times R_M}$
2. unfold($\mathcal{D}^d$, d) as $\mathbf{D}_{(d)} \in \mathbb{R}^{N_d \times (\prod_{k \ne d} R_k)}$
3. Construct variance matrix $\mathbf{C}_d = \mathbf{D}_{(d)}^T \mathbf{D}_{(d)}$
4. Compute $\mathbf{U}_d$ by diagonalizing $\mathbf{C}_d$

```
Input:
The tensor window $\mathcal{D} \in \mathbb{R}^{W\times N_1\times\cdots\times N_M}$
The dimensionality of the output tensors $\mathcal{Y} \in \mathbb{R}^{R_0\times\cdots\times R_M}$.
Output:
The projection matrix $\mathbf{U}_0 \in \mathbb{R}^{W\times R_0}$, $\mathbf{U}_i|_{i=1}^{M} \in \mathbb{R}^{N_i\times R_i}$
and the core tensor $\mathcal{Y}$.
Algorithm:
1. Initialize $\mathbf{U}_i \,|_{i=0}^{M}$
2. Conduct 3 - 5 iteratively
3. For $k = 0$ to $M$
4.    Fix $\mathbf{U}_i$ for $i \neq k$ and find the $\mathbf{U}_k$ that
      minimizes $d(\mathcal{D}, \mathcal{D}\prod_{i=0}^{M} \times_i (\mathbf{U}_i\mathbf{U}_i^T))$
5. Check convergence
6. Calculate the core tensor $\mathcal{Y} = \mathcal{D}\prod_{i=0}^{M} \times_i \mathbf{U}_i$
```

Figure 2. Iterative tensor analysis

Convergence checking: We use the standard approach of monitoring the change of the projection matrices until it is sufficiently small. Formally, the change on i-th mode is quantified by $\mathrm{trace}(\big||\mathbf{U}_{i,\text{new}}^T\mathbf{U}_{i,\text{old}}| - \mathbf{I}\big|)$.

3.2 Independent and moving window tensor analysis

Here we first introduce independent-window tensor analysis (IW) as the baseline algorithm. Then moving-window tensor analysis (MW) is presented that exploits the time dependence structure to quickly set a good initial condition, thereby significantly reducing the computational cost.

Independent window tensor analysis (IW)

IW is a simple way to deal with tensor windows by fitting the model independently. At every timestamp a tensor window $\mathcal{D}(n, W)$ is formed, which includes the current tensor $\mathcal{D}_n$ and $W - 1$ old tensors. Then we can apply the alternating least squares method (Figure 2) on $\mathcal{D}(n, W)$ to compute the projection matrices $\mathbf{U}_i|_{i=0}^{M}$. The projection matrices $\mathbf{U}_i$ can, in theory, be any orthonormal matrices. For instance, we initialize $\mathbf{U}_i$ to be a $N_i \times R_i$ truncated identity matrix in the experiment, which leads to extremely fast initialization of the projection matrices. However, the number of iterations until convergence can be large.

Moving-window tensor analysis (MW)

MW utilizes the overlapping information of two consecutive tensor windows to update variance matrices $\mathbf{C}_d|_{d=1}^{M}$. More specifically, given a tensor window $\mathcal{D}(n, W) \in \mathbb{R}^{W\times N_1\times\cdots\times N_M}$, we have $M+1$ variance matrices $\mathbf{C}_i\,|_{i=0}^{M}$, one for each mode. Note that the current window $\mathcal{D}(n, W)$ removes an old tensor $\mathcal{D}_{n-W}$ and includes a new tensor $\mathcal{D}_n$, compared to the previous window $\mathcal{D}(n-1, W)$.

Update modes 1 to M: For all but the time mode, the variance matrix is as follows:

$$\mathbf{C}_d^{old} = \begin{bmatrix}\mathbf{X}\\ \mathbf{D}\end{bmatrix}^T \begin{bmatrix}\mathbf{X}\\ \mathbf{D}\end{bmatrix} = \mathbf{X}^T\mathbf{X} + \mathbf{D}^T\mathbf{D}$$

where $\mathbf{X}$ is the unfolding matrix of the old tensor $\mathcal{D}_{n-W}$ and $\mathbf{D}$ is the unfolding matrix of tensor window $\mathcal{D}(n-1, W-1)$ (i.e., the overlapping part of the two consecutive tensor windows). Similarly, $\mathbf{C}_d^{new} = \mathbf{D}^T\mathbf{D} + \mathbf{Y}^T\mathbf{Y}$, where $\mathbf{Y}$ is the unfolding matrix of the new tensor $\mathcal{D}_n$. As a result, the update can be easily achieved as follows:

$$\mathbf{C}_d \leftarrow \mathbf{C}_d - \mathbf{D}_{n-W,(d)}^T\mathbf{D}_{n-W,(d)} + \mathbf{D}_{n,(d)}^T\mathbf{D}_{n,(d)}$$

where $\mathbf{D}_{n-W,(d)}(\mathbf{D}_{n,(d)})$ is the mode-d unfolding matrix of tensor $\mathcal{D}_{n-W}(\mathcal{D}_n)$. Intuitively, the variance matrix can be updated easily when adding or deleting rows from an unfolded matrix, since all computation only involves the added and deleted rows.

Updating time mode (mode 0) cannot be done efficiently as the other modes. Fortunately, the iterative algorithm actually only needs initialization for all but one mode in order to start. Therefore, after initialization of the other modes, the iterated update starts from the time mode and proceeds until convergence. This gives both quick convergence and fast initialization. The pseudo-code is listed in Figure 3.

```
Input:
The new tensor $\mathcal{D}_n \in \mathbb{R}^{N_1\times\cdots\times N_M}$ for inclusion
The old tensor window $\mathcal{D}_{n-W} \in \mathbb{R}^{N_1\times\cdots\times N_M}$ for removal
The old variance matrices $\mathbf{C}_d|_{d=1}^{M}$
The dimensionality of the output tensors $\mathcal{Y} \in \mathbb{R}^{R_0\times\cdots\times R_M}$
Output:
The new variance matrices $\mathbf{C}_d|_{d=1}^{M}$
The projection matrices $\mathbf{U}_i|_{i=0}^{M} \in \mathbb{R}^{N_i\times R_i}$
Algorithm:
 // Initialize every mode except time
 1. For $d = 1$ to $M$
 2.    Mode-$d$ matricize $\mathcal{D}_{n-W}(\mathcal{D}_n)$ as $\mathbf{D}_{n-W,(d)}(\mathbf{D}_{n,(d)})$
 4.    Update $\mathbf{C}_d \leftarrow \mathbf{C}_d - \mathbf{D}_{n-W,(d)}^T\mathbf{D}_{n-W,(d)} + \mathbf{D}_{n,(d)}^T\mathbf{D}_{n,(d)}$
 5.    Diagonalization $\mathbf{C}_d = \mathbf{U}_d\mathbf{\Lambda}_d\mathbf{U}_d^T$
 6.    Truncate $\mathbf{U}_d$ to first $R_d$ columns
 7. Apply the iterative algorithm with the new initialization
```

Figure 3. Moving-window tensor analysis (MW)

4 Performance Evaluation

Data Description `SENSOR`**:** The sensor data are collected from 52 MICA2 Mote sensors placed in a lab, over a period of a month. Every 30 seconds each Mote sensor sends to the central collector via wireless radio four types of measurements: light intensity, humidity, temperature, battery voltage. In order to compare different types, we scale each type of measurement into zero mean and unit variance. This calibration process can actually be done online since mean and variance can be easily updated incrementally. Note that we put equal weight on all measurements across all nodes. However, other weighting schemes can be easily applied, based on the application.

Characteristics: The data are very bursty but still correlated across locations and time. For example, the measurements of same type behave similarly, since all the nodes are deployed in the same lab.

Tensor construction: By scanning through the data once, the tensor windows are incrementally constructed and processed/decomposed. More specifically, every tensor window is a 3-mode tensor with dimensions (W, 52, 4) where W varies from 100 to 5000 in the experiments.

Parameters: This experiment has three parameters: 1) **Window size W**: the number of timestamps included in each tensor window. 2) **Step ratio S**: the number of newly arrived tensors in the new tensor windows divided by window size W (a ratio between 0 and 1). 3) **Core tensor size**: (R_0, R_1, R_2) where R_0 is the size of time mode. IW and MW reach the same error level[1] across all the experiments, since we use the same termination criterion for the iterative algorithm in both cases.

Stable over time: Figure 4(a) shows the CPU time as a function of elapsed time, where we set $W = 1000$, $S = .2$ (i.e. 20% new tensors). Overall, CPU time for both IW and MW exhibits a constant trend. MW achieves about 30% overall improvement compared to IW, on both datasets.

The performance gain of MW comes from its incremental initialization scheme. As shown in Figure 4(b), the CPU time is strongly correlated with the number of iterations. As a result of MW, which reduces the number of iterations needed, MW is much faster than IW.

Consistent across different parameter settings:

Window size: Figure 4(c) shows CPU time (in log-scale) vs. window size W. CPU time is increasing with window size. Note that the MW method achieves big computational saving across all sizes, compared to IW.

Step size: Figure 4(d) presents step size vs. CPU time. MW is much faster than IW across all settings, even when there is no overlap between two consecutive tensor windows (i.e., step size equals 1). This clearly shows that the importance of a good initialization for the iterative algorithm.

Core tensor size: We vary the core tensor size along the time-mode and show CPU time as a function of this size (see Figure 4(e)). Again, MW performs much faster than IW, over all sizes. Similar results are achieved when varying the sizes of the other modes, so we omit them for brevity.

[1]Reconstruction error is $e(\mathcal{D}) = \frac{||\mathcal{D}-\tilde{\mathcal{D}}||_F^2}{||\mathcal{D}||_F^2}$, where the tensor reconstruction is $\tilde{\mathcal{D}} = \mathcal{D}\prod \times_i (\mathbf{U}_i\mathbf{U}_i^T)$

5 Application and Case study

In this section, we introduce a powerful mining application of window-based tensor analysis, *Multi-Aspect Correlation Analysis* (MACA). Then we present a case study of MACA on the SENSOR dataset. Figure 5 shows two factors

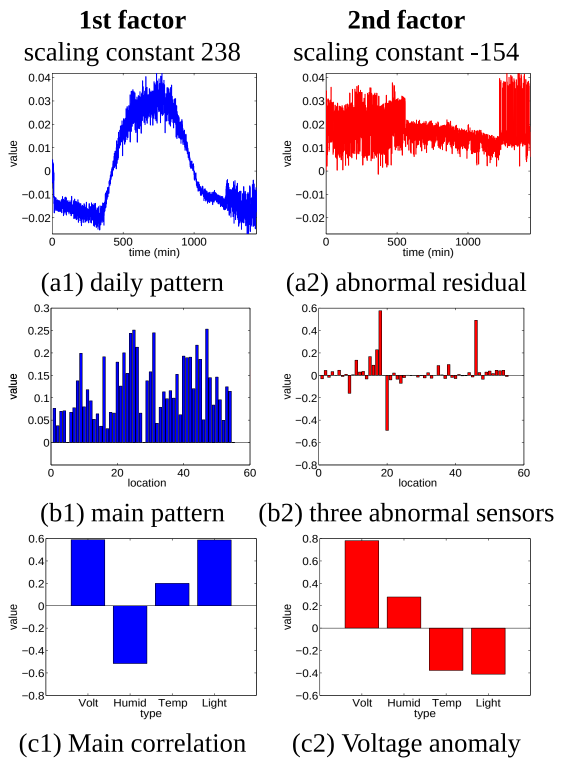
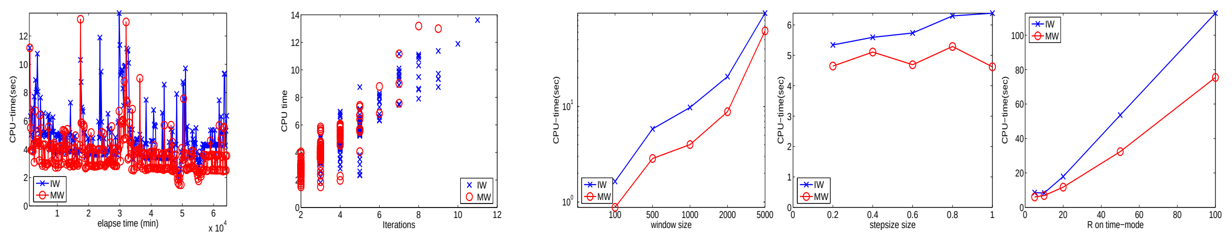


Figure 5. Case study on environmental data

across three aspects (modes): time, location, and type. Intuitively, each factor corresponds to a trend and consists of a global "importance" weight of this trend, a pattern across time summarizing the "typical behavior" for this trend and, finally, one set of weights for each aspect (representing their participation in this trend). More specifically, the trends are from projection matrices and the weights from core tensors.

Figures 5(a1,b1,c1) show the three components (one for each aspect) of the first factor, which is the main trend. If we read the components independently, Figure 5 (a1) shows the daily periodic pattern over time (high activation during the day, low activation at night); (b1) shows the participation weights of all sensor locations, where the weights seem to be uniformly spread out; (c1) shows that light, temperature and voltage are positively correlated (all positive values) while they are anti-correlated with humidity (negative



(a) time (b) num of iteration (c) window size (d) stepsize (e) rank

Figure 4. Experiments

value). All of those patterns can be confirmed from the raw data. All three components act together scaled by the constant factor 238 to form an approximation of the data.

Part of the residual of that approximation is shown as the second factor in Figures 5(a2,b2,c2), which corresponds to some major anomalies. More specifically, Figure 5 (a2) suggest that this pattern is uniform across all time, without a clear trend as in (a1); (b2) shows that two sensors have strong positive weights while one other sensor has a dominating low weight; (c2) tells us that this happened mainly at voltage which has the most prominent weight. The residual (i.e., the information remaining after the first factor is subtracted) is in turn approximated by combining all three components and scaling by constant -154. In layman terms, two sensors have abnormally low voltage compared to the rest; while one other sensor has unusually high voltage. Again all three cases are confirmed from the data.

6 Related work

Data streams have been extensively studied in recent years. Recent surveys [6] have discussed many data stream algorithms. Among them, the sliding (or moving) window is a popular model in data stream literature. Most of them monitor some statistics in the sliding window over a single stream using probabilistic counting techniques, while we work with a more general notion of multi-aspect streams.

Tensor algebra and multilinear analysis have been applied successfully in many domains. Powerful tools have been proposed, including the Tucker decomposition [9], parallel factor analysis [4] or canonical decomposition [2]. Kolda et al. [5] apply PARAFAC on web graphs to generalize the hub and authority scores for web ranking through term information. These methods usually assume static data, while we are interested in streams of tensors. For the dynamic case, Sun et. al [8] proposed dynamic and streaming tensor analysis for higher-order data streams, but the time aspect is not analyzed explicitly as in this paper.

7 Conclusion

In collections of multiple, time-evolving data streams from a large number of sources, different data values are often associated with multiple aspects. Usually, the patterns changes over time. In this paper, we propose the *tensor stream* model to capture the structured dynamics in such collections of streams. Furthermore, we introduce *tensor windows*, which naturally generalize the sliding window model. Under this data model, two techniques, independent and moving window tensor analysis (IW and MW), are presented to incrementally summarize the tensor stream. The summary consists of local patterns over time, which formally correspond to core tensors with the associated projection matrices. Finally, extensive performance evaluation and case study demonstrate the efficiency and effectiveness of the proposed methods.

8 Acknowledgement

We are pleased to acknowledge Brett Bader and Tamara Kolda from Sandia National lab for providing the tensor toolbox [1] which makes our implementation and experiments an easy job.

References

[1] B. W. Bader and T. G. Kolda. Matlab tensor toolbox version 2.0, http://csmr.ca.sandia.gov/ tgkolda/tensortoolbox/. 2006.

[2] J. D. Carroll and J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of 'eckart-young' decomposition. *Psychometrika*, 35(3), 1970.

[3] L. de Lathauwer. *Signal Processing Based on Multilinear Algebra*. PhD thesis, Katholieke, University of Leuven, Belgium, 1997.

[4] R. Harshman. Foundations of the parafac procedure: model and conditions for an explanatory multi-mode factor analysis. *UCLA working papers in phonetics*, 16, 1970.

[5] T. G. Kolda, B. W. Bader, and J. P. Kenny. Higher-order web link analysis using multilinear algebra. In *ICDM*, 2005.

[6] S. Muthukrishnan. *Data streams: algorithms and applications*, volume 1. Foundations and Trends. in Theoretical Computer Science, 2005.

[7] S. Papadimitriou, J. Sun, and C. Faloutsos. Streaming pattern discovery in multiple time-series. In *VLDB*, 2005.

[8] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: Dynamic tensor analysis. In *KDD*, 2006.

[9] L. R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3), 1966.