Tensor Decompositions for Graph Clustering

Michiel Vandecappelle, Martijn Boussé, *Student Member, IEEE*, Frederik Van Eeghem, *Student Member, IEEE*, and Lieven De Lathauwer, *Fellow, IEEE*

***Abstract*—Graph clustering methods typically only consider the direct connections between nodes of a graph. However, a better clustering can often be obtained by exploiting information about higher-order structures in the graph. For a social graph for example, one can focus on its triangles to obtain a better clustering. Just like the adjacency matrix holds information about direct connections in a graph, higher-order structure can be stored in a tensor. By computing a suitable tensor decomposition, one can then cluster the graph based on its higher-order structure. In this paper, we propose a new method for tensor-based graph clustering. We show how to tensorize graphs based on triangles and various other structures and we use the emerging concept of coupled tensor decompositions to take multiple structures into account simultaneously. Experiments on synthetic and real-life graphs show that the proposed methods compute clusterings that can adequately capture the requested higher-order structures.**

I. Introduction

GRAPHS form an abstraction of networks by reducing them to a number of entries that are either connected or not [1]. They can be used to represent networks in many different contexts, e.g. social networks, computer networks and supply networks [2]. Clustering algorithms try to partition a graph into several subgraphs, called clusters, so that the nodes of a cluster are relatively well connected compared to the rest of the graph, while the number of edges between these clusters is limited. Many different clustering algorithms have been designed, depending on the specific structure of the graph and the goal of the clustering [2][3]. A common trait of many of these methods is that they only consider the direct edges of a graph. This is also referred to as the second-order structure of the graph because only connections between two nodes are taken into account. While the results of these second-order methods are often very good, there are cases where the structure of a graph is better characterized by its higher-order structures [4]. These higher-order structures are connections between more than two nodes in the graph, such as triangles, rectangles or stars. Ignoring higher-order structures while clustering a graph, can lead to a clustering that fails to capture the underlying structure of the graph. For instance, ignoring triangles while clustering social graphs leads to clusters that do not adequately model real-world communities [5]. For example: in friend networks, triangles are unusually prevalent. As friends of friends are often friends themselves, the triangle structure of the graph should definitely be taken into account during the clustering.

The edge structure of a graph can be stored compactly in the adjacency matrix [1]. While matrices are well-suited to represent this second-order information, they cannot be used to represent higher-order structures in a graph directly. For example, to represent a triangle in a graph, which is a connection between three nodes of the graph, we would ideally want to use three indices and a matrix only provides two. To alleviate this restriction to second-order structures, tensors can be used. Tensors are higher-order extensions of vectors and matrices [6], [7]. They allow the representation of data in an arbitrary number of dimensions, while vectors and matrices can only represent data in one and two dimensions, respectively. Several tensor decompositions have been proposed to analyze higher-order data. The fact that these decompositions have mild uniqueness properties, is one of the reasons why tensors are so powerful. Tensors were first applied in the analysis of psychometric data and have since been used in many different fields, such as chemometrics [8], medical imaging [9], signal processing [7], blind source separation [7] and data mining [10]. In the context of graphs, they can be used to represent arbitrary higher-order structures of the graph.

There exist several clustering algorithms that use a tensor representation of a graph. Most of these consider time-evolved or layered graphs, such as when websites are linked through a number of different keywords [11] or authors are linked through either citations, collaborations or associations [12]. Using tensors to represent higher-order structures in graphs, however, is still relatively unexplored. Benson et al. propose a method that uses a stochastic triangle tensor [13]. Their tensor represents a randomized walk on the graph that, after visiting two nodes of the graph, uniformly chooses the next node from the nodes that form a triangle with these two nodes. They then recursively perform a bisection of the graph until the requested number of clusters is reached, cutting as few triangles as possible during this process.

We propose a general higher-order framework for clustering graphs that can not only handle triangle structures, but also more complex higher-order structures. First, we will define the triangle tensor which holds information about the triangles in a graph. We then show that a decomposition of this tensor can

Manuscript received November 1, 2016.

This research is funded by (1) Research Council KU Leuven: C1 project c16/15/059-nD and CoE PFV/10/002 (OPTEC), (2) F.W.O.: project G.0830.14N and G.0881.14N, (3) the Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO II, Dynamical systems, control and optimization, 2012-2017), (4) EU: The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Advanced Grant: BIOTENSORS (no. 339804). This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information.

The authors are with the Department of Electrical Engineering (ESAT), KU Leuven, Kasteelpark Arenberg 10, B-3001 Leuven, Belgium. (e-mail: michiel.vandecappelle@kuleuven.be; martijn.boussé@kuleuven.be; frederik.vaneeghem@kuleuven.be; lieven.delathauwer@kuleuven.be).

Michiel Vandecappelle, Frederik Van Eeghem and Lieven the Lathauwer are with the Group of Science, Engineering and Technology, KU Leuven Kulak, E. Sabbelaan 53, B-8500 Kortrijk, Belgium.

reveal a good clustering of the graph, as the different terms of this decomposition correspond to the different clusters of the graph. Next, we will extend this method by showing that arbitrary higher-order structures, such as for instance cliques and stars, can be modeled by constructing suitable tensorizations of the graph. Additionally, coupled decompositions are used to consider multiple higher-order structures at once.

In Section II, we introduce notation and basic definitions for tensors and graphs. In Section III, a tensor-based graph clustering method is proposed that focuses on the triangle structure of the graph. Tensorizations for other higher-order structures are then discussed in Section IV. In Section V, we outline how decompositions of different tensorizations can be coupled to combine multiple types of higher-order information. Numerical experiments are performed in Section VI on synthetic and real-life data. Finally, we conclude in Section VII.

II. Notation and basic definitions

We first introduce some notation and relevant higher-order operations for tensors. Next, we define the canonical polyadic decomposition (CPD) [14], [15] and give a short overview of relevant graph concepts.

A. Notation and operations

Scalars, vectors and matrices will be denoted by lowercase (a), bold lowercase ($\mathbf{a}$) and bold uppercase letters ($\mathbf{A}$), respectively. Vector and matrix entries are written as a_i and a_{ij}, respectively. The nth element of a sequence of matrices is denoted by $\mathbf{A}^{(n)}$. We will refer to tensors by using letters in calligraphic script ($\mathcal{T}$). Entries of an Nth-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ are denoted by $t_{i_1 \cdots i_N}$. The outer product of N vectors, denoted by $\mathbf{v}^{(1)} \otimes \cdots \otimes \mathbf{v}^{(N)}$, is a natural extension of the outer product of two vectors. The result is an Nth-order tensor $\mathcal{T}$ of which each entry is defined as follows: $t_{i_1 \ldots i_N} = v^{(1)}_{i_1} v^{(2)}_{i_2} \cdots v^{(N)}_{i_N}$.

B. Decomposition of tensors

In order to define the rank of a tensor, we first need to define rank-1 tensors. An Nth-order tensor is called a rank-1 tensor if it is the outer product of N non-zero vectors. The rank of a tensor is then the minimal number of terms that is needed to write the tensor as a linear combination of rank-1 tensors. These concepts can now be used to define the CPD.

Definition 1 (*Canonical polyadic decomposition (CPD)*)**.** *The CPD of an Nth-order rank-R tensor $\mathcal{T}$ decomposes the tensor as a linear combination of R rank-1 terms:*

$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(N)}.$$

The vectors $\mathbf{u}_r^{(n)}$ are usually collected into N factor matrices $\mathbf{U}^{(n)}$ as follows: $\mathbf{U}^{(n)} = \begin{bmatrix} \mathbf{u}_1^{(n)} \ldots \mathbf{u}_R^{(n)} \end{bmatrix}$. The CPD of an Nth-order tensor $\mathcal{T}$ can then be written as $\mathcal{T} = [\![\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}]\!]$.

A big advantage of the CPD in comparison to matrix decompositions is that it is essentially unique under fairly mild conditions. If changing the order of the rank-1 terms and scaling and counter-scaling within the same term are the only indeterminacies, the CPD is called essentially unique. For uniqueness properties of the CPD, see [16]–[21] and references therein.

C. Graphs

We very briefly list some relevant graph concepts:

Definition 2 (*Graph*)**.** *A graph $G(V, E)$ of order N consists of a set of nodes $V = \{v_1, ..., v_N\}$ and a set of edges $E = \{e_1, ..., e_M\}$. Edges form pairwise connections between the nodes of the graph. An edge between nodes v_i and v_j is denoted by (v_i, v_j).*

Information about the edge structure of a graph is usually collected in an adjacency matrix. We will use a modified version of this matrix, where its diagonal is set to one:

Definition 3 (*Modified adjacency matrix*)**.** *An $(N \times N)$-modified adjacency matrix $\mathbf{A}$ is a matrix representation of a graph $G(V, E)$ of order N with*

$$\begin{cases} a_{ij} = 1 & \text{if } v_i = v_j \text{ or } v_i \text{ and } v_j \text{ share an edge,} \\ a_{ij} = 0 & \text{otherwise.} \end{cases}$$

III. Tensor-based graph clustering

We derive a tensor-based method for graph clustering that exploits triangle structure via the triangle tensor of a graph. We will cluster the graph by computing a CPD of the triangle tensor. The rank R of the CPD equals the number of clusters and each rank-1 term corresponds to one of the clusters of the graph. The main idea behind our method is illustrated in Figure 1.

A. Triangle tensor

Triangles form natural higher-order extensions of edges in graphs. They hold a lot of structural information about the graph. While two nodes are maximally connected when they are connected by an edge, three nodes are only maximally connected if they form a triangle. Because of this, triangles tend to occur more in denser parts of graphs. For triangle-heavy graphs, such as social graphs [22], it can be very useful to locate the parts of the graph that have the highest triangle count and to build up the clustering from these parts on [23]. To store information about the triangles in a graph, we will define the triangle tensor as a generalization of the adjacency matrix. The entries of this third-order tensor have the value one if the three nodes that correspond to the entry form a triangle:

Definition 4 (*Triangle tensor of a graph*)**.** *The triangle tensor $\mathcal{X}$ of an N-node graph is an $(N \times N \times N)$ tensor in which the entries $x_{\sigma(ijk)}$ have the value one if the nodes v_i, v_j and v_k form a triangle in the graph. $\sigma(ijk)$ denotes all permutations of the indices i,j, and k. The entries of the type x_{iij} have the*

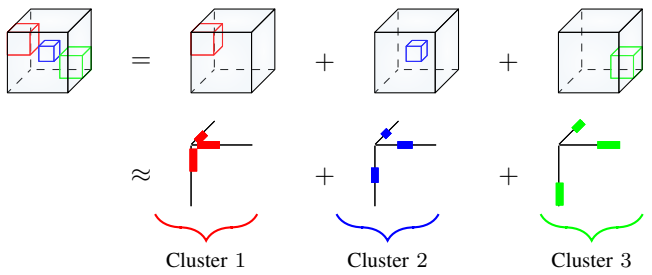

Figure 1. CPD of a block diagonal triangle tensor. A triangle tensor is constructed for a graph with separated clusters, leading to a block diagonal tensor (after a possible reordering of the nodes). This tensor consists of three blocks and is approximated by a sum of three rank-1 tensors. Each rank-1 tensor corresponds to one block of the tensor, so only the highlighted values are non-zero. The three clusters of the graph can thus easily be extracted from the three rank-1 tensors by locating the non-zero values of these tensors, as these non-zero values correspond to the indices of the nodes that are in the cluster.

$$\mathbf{U} = \begin{bmatrix} \mathbf{1} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{1} & 0 & \cdots & 0 & 0 \\ 0 & \mathbf{1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \mathbf{1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \mathbf{1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & \mathbf{1} \end{bmatrix} \begin{matrix} \} n_1 \\ \\ \} n_2 \\ \\ \\ \} n_R \\ \\ \end{matrix}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad}_{R}$$

Figure 2. A factor matrix for the CPD of the triangle tensor of a graph.

value one if the nodes v_i and v_j are connected. All entries v_{iii} are set to one.

The triangle tensor is binary and symmetric and often very sparse, because triangles are relatively uncommon in many graphs. Because of this, it can be stored compactly.

B. Clustering

By computing a CPD of its triangle tensor, a graph can be clustered. We consider as the ideal situation that the clusters of the graph are completely separated and fully connected, meaning that all nodes are connected to all others nodes in the cluster, but not to any of the nodes in other clusters. In this case, the N nodes of the graph can be ordered such that a block diagonal triangle tensor is obtained, as shown in Figure 1. Each of the R blocks of this tensor then consists of only ones and corresponds to one of the fully connected clusters of the graph. Due to its specific structure, the triangle tensor is in this case a rank-R tensor. Each rank-1 term corresponds to one of the blocks of the tensor and its entries will either have the value zero or one if we subsequently compute a rank-R CPD of this block diagonal triangle tensor. The triangle tensor is a non-negative symmetric tensor, so its CPD will be non-negative and symmetric as well, which in turn implies that all the factor matrices are equal. The rank-R CPD of the triangle tensor $\mathcal{X}$ will then look as follows:

$$\mathcal{X} = [\![\mathbf{U}, \mathbf{U}, \mathbf{U}]\!]_R \,,$$

with $\mathbf{U} \in \mathbb{R}^{N \times R}$. In Figure 2, a factor matrix of this decomposition is shown. The n_i entries corresponding to each of the R clusters $\mathcal{C}_i$ are marked. One can see that the different clusters of the graph can easily be extracted, even if the nodes are not reordered: all nodes that are part of the i-th cluster have the value one in the i-th column and zeros in the other columns. The fact that the CPD reveals the correct clustering in the ideal case, is the basis of our clustering method. This property is summarized in the following theorem:

Theorem 1 (CPD of the triangle tensorization)**.** *For a graph with R fully connected and perfectly separated clusters, the triangle tensor $\mathcal{X}$ has rank R and the factor matrices of the CPD of $\mathcal{X}$ indicate the clustering.*

Proof. Every fully connected cluster $\mathcal{C}_i$ of the graph corresponds (after reordering of the nodes) to an $(n_i \times n_i)$-block of the tensor consisting of only ones, which we denote by $\mathcal{B}_i$. Such a block is a rank-1 tensor, because it can be decomposed as $\mathcal{B}_i = \mathbf{1}_{n_i} \otimes \mathbf{1}_{n_i} \otimes \mathbf{1}_{n_i}$, with $\mathbf{1}_{n_i}$ a vector of length n_i consisting of only ones. By summing these R rank-1 terms, one obtains a rank-R decomposition of the triangle tensor $\mathcal{X}$. According to rank conditions given by Kruskal [17], the tensor rank is at least the rank of the space spanned by the mode-n slices of the tensor. For the tensor $\mathcal{X}$, the rank of this space is R, so the tensor rank is R and the decomposition is a CPD. Moreover, it is also unique, as the k-rank of each factor matrix $\mathbf{U}^{(n)}$ ($k_{\mathbf{U}^{(n)}}$) is equal to R and the tensor thus fulfills the uniqueness condition $k_{\mathbf{U}^{(1)}} + k_{\mathbf{U}^{(2)}} + k_{\mathbf{U}^{(3)}} \geq 2R+2$ for $R > 1$ [24]. The CPD of the triangle tensor, $\mathcal{X} = [\![\mathbf{U}, \mathbf{U}, \mathbf{U}]\!]_R$, is thus always identical (after reordering) to the decomposition that we constructed, so it corresponds to the correct clustering of the graph. □

The CPD only reveals the correct clustering directly in the ideal case. We will argue below, however, that our approach can be applied to arbitrary graphs by exploiting the specific structure of the factor matrices.

If the clusters of the graph are completely separated, but not fully connected, then the triangle tensor of the graph is still a block diagonal tensor, but these blocks no longer consist of only ones. As a result, a triangle tensor with R blocks has rank $R' \geq R$. However, we can still compute a rank-R CPD approximation:

$$\mathcal{X} \approx [\![\mathbf{U}, \mathbf{U}, \mathbf{U}]\!]_R \,.$$

The R rank-1 terms are expected to correspond reasonably well to the R different clusters of the graph if the clusters are sufficiently densely connected and do not differ too much in size. The factor matrices now contain non-zero values different from one, but the clusters can still easily be found by locating the non-zero values in the factor matrices.

In a more realistic situation, there will be connections between the different clusters of the graph. In that case, the

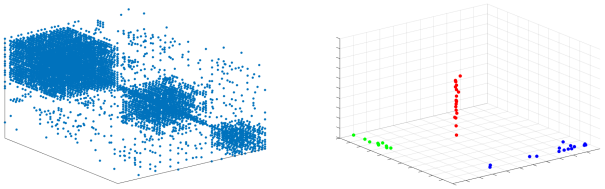

Figure 3. A triangle-heavy graph is clustered into three clusters using the CPD of the triangle tensor. A spy plot of the triangle tensor (left) reveals that its block-diagonal structure is dominant. As a result, by applying a clustering algorithm to the rows of the factor matrices, one can still easily distinguish the three clusters (right).

block-diagonal structure of the triangle tensor is lost. However, this block-diagonal structure remains dominant as long as the inter-cluster connections do not introduce too many new triangles, as shown in Section VI. Of course, the number of non-zero entries in the factor matrices is now also higher and the different clusters of the graph cannot be extracted from the factor matrices as easily as before. However, one may still obtain a good clustering of the graph by using a suitable data clustering algorithm (k-means++, k-harmonic means, hierarchical clustering) on the rows of the factor matrix: rows of nodes in the first cluster can still be separated from rows of nodes in the second cluster if the number of inter-cluster connections is not too high. This is illustrated in Figure 3.

The tensor-based clustering method builds up its clusters around triangle-heavy subgraphs of the graph. These subgraphs each form the basis of a cluster and lead to the (after reordering) dominant block-diagonal structure of the triangle tensor. The method is thus particularly suited to cluster graphs where the graph structure follows the triangle distribution. For some graphs, the clustering can be influenced by nodes that are part of no triangles. By using the triangle tensor, the negative influence of these nodes can already be greatly reduced. If necessary, an additional denoising step can be applied before the graph is clustered. This is done by trimming the triangle tensor and only clustering nodes that are part of a certain number of triangles with the CPD approach. The other nodes are then added to the cluster that they are most connected to. In this way, the triangle-heavy clusters of the graph can be found without the negative influence of the loosely connected nodes. The full clustering algorithm is as follows.

Algorithm 1: Tensor-based graph clustering

Input : Graph, number of clusters k.
Output: k clusters of the graph.

1. Construct the triangle tensor of the graph.
2. (Optional) Trim the triangle tensor to perform a denoising.
3. Compute a rank-k CPD approximation of the tensor.
4. Cluster nodes using the factor matrices.
5. (Optional) Assign the loosely-connected nodes to the cluster they are most connected to.

IV. Higher-order structures

So far we have only discussed graphs with a triangle structure. However, our approach allows us to exploit other higher-order structures in graphs, such as cliques and stars. Those higher-order structures can be encoded in higher-order tensors similar to the triangle tensor. A tensorization [25] for an Nth-order structure can straightforwardly be defined as an Nth-order tensor for which the entries have the value one if the corresponding nodes form the requested structure. More efficient tensorizations, i.e., representing an Nth-order structure with a tensor of order smaller than N, are in some cases also possible. A clustering that is obtained from these alternative tensorizations has clusters that are based on the requested higher-order structures instead of triangles. The use of specific tensorizations is especially useful if information is available about the types of clusters that one expects to see in the graph, e.g., triangles for most social graphs and stars for telephone networks.

A good tensorization should lead to a block-diagonal tensor for a graph with separated clusters. For clusters that perfectly form the requested structure, these R blocks should consist of only ones. Hence, in analogy with Theorem 1, for a graph with R fully connected and perfectly separated clusters and a tensorization $\mathcal{X}$ that has rank R, the factor matrices of the CPD of $\mathcal{X}$ can be used to cluster the graph.

Higher-order structures can reveal more detailed information about a graph than lower-order structures. For instance, knowing that four nodes form a rectangle is more informative than knowing that four connections exist between those four nodes. Higher-order tensors can thus store more information about a graph, but storage requirements and computation time rise exponentially if the order of a tensor is increased. Increasing the number of rows and columns of a matrix leads to a quadratic growth of the number of entries, but for higher-order tensors, this growth is cubic or faster. Several strategies have been developed to counter this 'curse of dimensionality' [26], such as using randomized block sampling [27] or incomplete tensors [28]. Additionally, tensors for higher-order structures are often very sparse. This can be exploited by only storing the nonzero values of the tensor instead of the full tensor. The use of higher-order information should, however, always be weighted against the additional complexity, computation, and storage requirements.

Below, we explore two interesting higher-order structures, namely an Nth-order clique tensor and a star tensor.

A. Cliques

Cliques are fully connected subgraphs of a graph of which the triangle tensor is the third-order variant. Third-, fourth- and fifth-order cliques are displayed in Figure 4. They are mainly useful to cluster triangle-heavy graphs, as higher-order cliques consist of many triangles and are consequently rarer than triangles. Clique tensors can be constructed analogously to the triangle tensor by simply increasing the order of the tensor.

Definition 5. *The Nth-order clique tensor $\mathcal{T}$ of a graph is an Nth-order tensor $\mathcal{T}$ of which the entries $t_{i_1 \dots i_N}$ have the value*

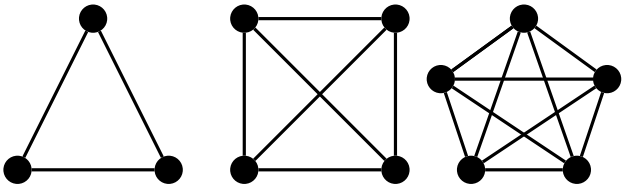

Figure 4. A third-order (i.e. a triangle) (left), fourth-order (center), and fifth-order (right) clique. These structures form maximally connected subgraphs of a graph. It is often a good choice to build the clusters of a clustering around these higher-order structures.

one if the nodes $v_{i_1}, \ldots, v_{i_N}$ form a clique. The entries with repeated indices have the value one if the respective nodes form a (lower-order) clique.

An Nth-order clique tensor holds information about connected N-tuples of nodes in a graph. As these nodes are maximally connected, they often correspond to important sections of a network where there is a lot of interaction between the nodes. Clusters that are built up around these higher-order cliques are quite robust, as higher-order cliques are unlikely to occur 'accidentally' due to their high number of edges. Increasing the order of the clique tensor can thus improve the quality of the clustering: one first clusters nodes that are part of high-order cliques and then gradually decreases the clique order to cluster the less connected nodes. This approach essentially extends the denoising step that was introduced above: first nodes are clustered that are part of a certain number of Nth order cliques, then those in $(N-1)$-order cliques and so on until only the nodes that are not part of any triangle are left. As higher-order clique tensors are often very sparse, they allow tensors of a relatively high order to be used if the sparseness is exploited.

The requirement that nodes have to form a complete clique can be relaxed. One can for example allow one (or a few) edge(s) to be missing in a clique, while the nodes still have the value one in the clique tensor. As such, a missing edge here and there does not influence the clustering.

B. Stars

Clique tensors are very useful for clustering triangle-heavy graphs, but not all graphs have such a structure. In centralized graphs, for example, certain select nodes are connected to many other nodes, but very few triangles are present. The star tensor is much more useful to display the structure of these graphs than the triangle tensor. Star tensors of arbitrary order can be defined, depending on the requested number of legs of the star structure. These star tensors can then be used to cluster a graph in such a way that the clusters are constructed around its most star-like subgraphs. We define the fourth-order star tensor, but higher-order star tensors can be constructed analogously. Some examples are given in Figure 5.

Definition 6 (*Fourth-order star tensor*)**.** *The fourth-order star tensor $\mathcal{T}$ of an Nth-order graph is an $(N \times N \times N \times N)$ tensor in which the entries t_{ijkl} and their permutations have the value one if at least one of the nodes v_i, v_j, v_k and v_l is connected to the three other nodes by a direct edge. The entries of the type t_{iijk} have the value one if one of the nodes v_i, v_j or v_k is connected to both others. The entries of the type t_{iiij} and t_{iijj} have the value one if the nodes v_i and v_j are connected. All entries t_{iiii} are set to one.*

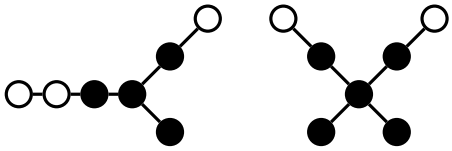

Figure 5. Two examples of star structures. For the graph on the left, the marked nodes form a fourth-order star structure. For the graph on the right, the marked nodes show a fifth-order star structure.

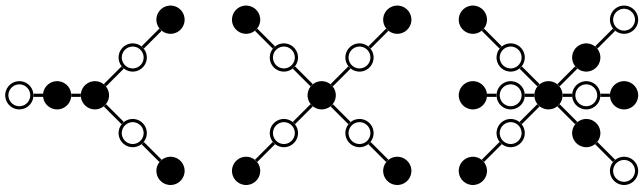

Figure 6. Fourth-order (left), fifth-order (center), and seventh-order (right) star structures with leg length 2. The highlighted nodes are examples of node tuples that have the value one in the extended star tensor, but not in the basic star tensor: in the extended star tensor, nodes do not have to be directly connected to the central node.

The star tensor can be adapted to localize star structures with longer legs in a graph without increasing the order of the tensor. This can be done by counting nodes located further along the legs of the stars as being connected to the central node: nodes that form a Nth-order star with leg length $\leq l$ all have the value one for their corresponding tensor entries. Some examples are shown in Figure 6. Using this extended star tensor, stars with longer legs can be represented efficiently. Additionally, central nodes of longer-legged stars are often very important nodes in the graph. With this modified star tensor, they are given more weight in a clustering, as stars introduce more non-zero values into this tensor if they have longer legs.

V. Coupled and structured decompositions

Recently, coupled tensor decompositions have received a lot of attention [29], [30]. These decompositions force certain terms, such as (parts of) factor matrices, to be identical in the decompositions of different tensors. In this way, features that are shared by the data tensors, e.g. common sources, can be extracted jointly for all tensors. For properties, algorithms and applications, see [7], [31]–[33] and references therein.

Coupled tensor decompositions can also be used for graph clustering. Different tensorizations of a graph can be coupled to exploit multiple higher-order structures of the graph. For example, the triangle tensor can be coupled to the modified adjacency matrix from Definition 3 to include both second- and third-order information about the graph. A symmetric CPD of the triangle tensor $\mathcal{T}$ and adjacency matrix $\mathbf{A}$ is computed simultaneously, while the factor matrix $\mathbf{U}$ is shared between

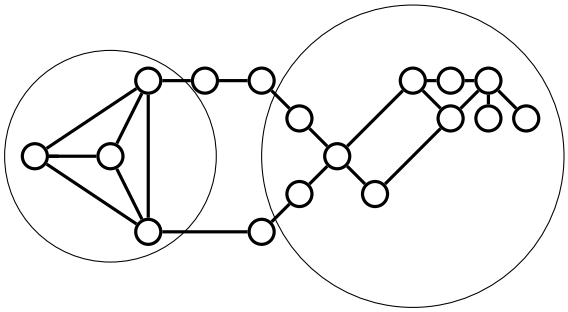

Figure 7. Example of a graph for which a coupled decomposition of the adjacency matrix, the triangle tensor and a star tensor is suitable. The graph has a triangle-heavy cluster (left), a star-heavy cluster (right) and a few nodes that are not part of any triangle or star.

both CPDs:

$$\mathbf{A} = \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{u}_r,$$
$$\mathcal{T} = \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{u}_r \otimes \mathbf{u}_r.$$

Adding the adjacency matrix to the decomposition is especially useful to cluster the nodes of the graph that are part of relatively few triangles, as these can not always be correctly assigned to the correct cluster using the triangle tensor alone.

As a second example, graphs with both star- and triangle-heavy clusters can be clustered by computing a coupled decomposition of a star tensor $\mathcal{S}$, the triangle tensor $\mathcal{T}$ and the adjacency matrix $\mathbf{A}$ of the graph:

$$\mathbf{A} = \sum_{r=1}^{R} \mathbf{u}_r \otimes \mathbf{u}_r,$$
$$\mathcal{T} = \sum_{r=1}^{R} (\mathbf{w}_T * \mathbf{u}_r) \otimes (\mathbf{w}_T * \mathbf{u}_r) \otimes (\mathbf{w}_T * \mathbf{u}_r),$$
$$\mathcal{S} = \sum_{r=1}^{R} (\mathbf{w}_S * \mathbf{u}_r) \otimes (\mathbf{w}_S * \mathbf{u}_r) \otimes (\mathbf{w}_S * \mathbf{u}_r) \otimes (\mathbf{w}_S * \mathbf{u}_r).$$

With this coupled decomposition, multiple types of clusters can be taken into account in the computation of the CPD approximation. The binary weight vectors $\mathbf{w}_T$ and $\mathbf{w}_S$ are multiplied element-wise $(*)$ with the vector $\mathbf{u}_r$ and are used to exclude nodes that do not belong to triangles or stars, respectively, from the rank-1 terms. In this way, clusters without triangles do not influence the star-heavy clusters and vice versa. Additionally, the different higher-order structures can also be weighted depending on their relative importance in the clustering. An example of a graph that benefits from this approach is given in Figure 7. This graph has a triangle-heavy cluster and a star-heavy cluster and a few nodes that are not part of any triangle or star. To assign these loosely-connected nodes to the other clusters, an iterative approach can be used: one assigns the nodes to the clusters they have the most connections with and repeats this process until all nodes have been assigned to a cluster.

VI. Numerical experiments and applications

We illustrate the proposed method for some synthetic and real-life graphs. First, we compare our method for triangles with the triangle method of Benson et al. [13] and a spectral method. Next, suitable graphs are clustered with the clique and the star tensors and a coupled decomposition is used to cluster a graph with different types of higher-order structures. Finally, we apply our method to a real-life social network.

Different approaches are possible to compute a clustering from the factor matrices of the CPD of a tensorization. We mainly used the k-means++ algorithm, but in some cases, mostly for uncoupled CPDs, better results were obtained by simply assigning each node to the cluster that had the largest entry in the factor matrix of the CPD.

Several measures have been proposed to assess the quality of a clustering. When the correct clustering is known, an exact measure, such as the Adjusted Rand Index (ARI) [34], which expresses the similarity between the computed and the correct clustering, can be used. Otherwise, a widely-used measure is the conductance of a graph [35]. The conductance rates the clustering quality by comparing the number of edges that are present within the clusters of a graph to the number of edges between its clusters. As we are interested in finding clusters with particular higher-order structures, however, this measure is not always good choice, e.g., a method that cuts trough a star structure only cuts one edge, which does not influence the conductance much, but destroys the requested star structure of the clusters. Instead, we use a generalized version of the conductance, called the motif conductance, which compares the number of higher-order structures within the clusters to those that are split by the clustering. This measure was first proposed for triangles [13] and has recently been extended to arbitrary higher-order structures [36].

Definition 7 (*Motif conductance*)**.** *The motif conductance* Φ *of a clustering is defined as*

$$\Phi_M(S) = cut_M(S, \overline{S}) / \min[vol_M(S), vol_M(\overline{S})],$$

with S *and* $\overline{S}$ *forming a partition of the nodes in two clusters,* cut_M *counting the nodes in higher-order structures* M *that are cut by the clustering and* vol_M *counting the nodes of higher-order structures that lie completely within a cluster.*

For a k-way clustering, the motif conductance is summed for all clusters. The motif conductance is well suited for our purposes, as it measures exactly what we want to accomplish: we try to form clusters that contain the higher-order structures of our choice and do not cut through them. Depending on the type of tensorization, the motif conductance is a triangle conductance, star conductance, etc.

In our synthetic experiments, many graphs were constructed by introducing connections between disconnected subgraphs. This was done by randomly flipping edges between clusters, i.e., introducing an edge where there previously was none or removing an existing edge between clusters. The number of reported flipped edges may exceed the number of interconnecting edges, because some edges might have been flipped more than once.

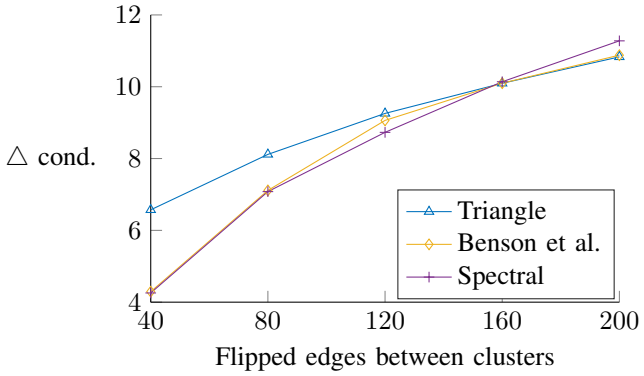

Figure 8. Triangle conductance for three different clustering methods. The triangle method has a higher conductance for graphs with loosely connected clusters, but as the clusters become more interconnected, our method outperforms the spectral method.

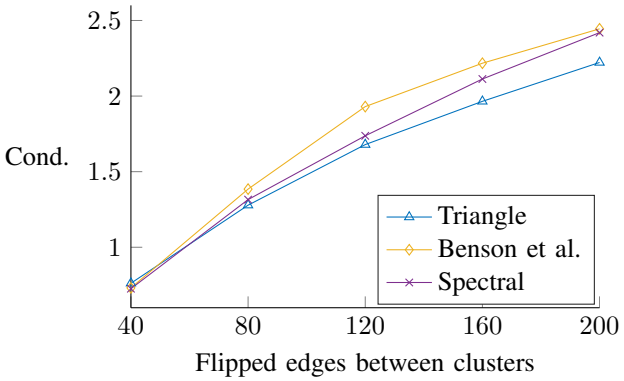

Figure 9. Conductance of the clusterings of three different clustering methods. The triangle method achieves a lower conductance, especially when the graphs are strongly interconnected.

A. Triangles

We compare our method with the triangle method of Benson et al. [13] and a spectral method on triangle-heavy graphs. The number of flipped edges between the clusters is varied to obtain different degrees of difficulty for the clustering. We report the average triangle conductance, normal conductance and ARI across 50 experiments per edge flip level on synthetic graphs with five triangle-heavy clusters of ten nodes in Figure 8, Figure 9 and Figure 10, respectively. All clusters consisted of eight random triangles and if a node was not part of any triangle, an extra triangle was added. The triangle method performs better than the other methods when the number of intercluster connections is high. As this method exploits the triangle structure in the graph, it is more robust against weaker intercluster connections. Our method manages to limit the number of triangles that are cut by the clustering and thus recovers the original clusters better than the other methods.

B. Cliques

The clique tensor can be used in a hierarchical clustering strategy, as mentioned above. First, we cluster nodes of higher-order cliques and then gradually decrease the order of the

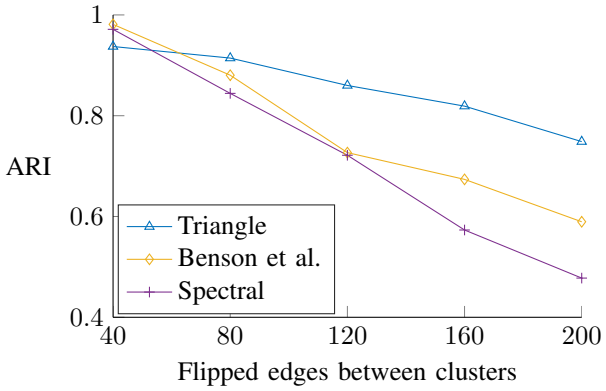

Figure 10. ARI for three different clustering methods. The triangle method outperforms the other methods for these graphs.

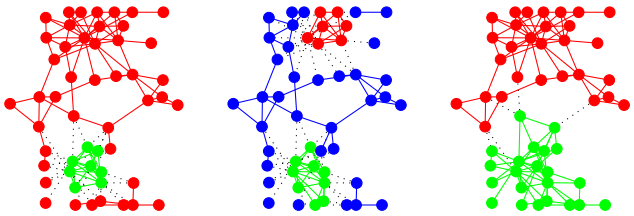

Figure 11. Using only the CPD of the triangle tensor (left), the k-means++ algorithm divides the graph into a very dense cluster (green) and a rest cluster (red). With a step-wise approach, the result is much better. This method first clusters the nodes that are part of 4th-order cliques (red and green) with a clique tensor (center) and leaves the other nodes unclustered (blue). Next, the nodes that are part of a triangle are clustered with the triangle tensor. Finally the remaining nodes are assigned to the existing clusters (red and green) using the adjacency matrix (right).

clique to cluster the remaining nodes. In the first step, we want to extract k rank-1 terms from the full clique tensor and leave the other nodes unclustered. There are two ways to achieve this result. One can trim the clique tensor to remove the nodes that are not part of a clique before the clustering and only cluster the remaining nodes into k clusters. Alternatively, one can decompose the clique tensor into $k+1$ instead of k terms. In this case. k rank-1 terms will each correspond to a clique of the graph and the extra term will hold all other nodes. The k clique terms can then be used to form the clustering. This step-wise approach is illustrated in Figure 11.

C. Stars

The average star conductance is computed across 100 experiments on synthetic graphs with five star-heavy clusters of ten nodes. In each cluster, ten stars were formed randomly. If a node remained unconnected, an extra star was added. Connections were added between the clusters by flipping 75 edges. Results are shown for the star conductance, the normal conductance and the ARI in Table I. The star method performs slightly better than the other methods in terms of star conductance: few stars are cut by the clustering compared to the number of stars in the clusters. Our triangle method has the lowest conductance score, but this score is achieved by cutting more stars than the other methods. This is not surprising, as the triangle method aims to cut as few triangles as possible

Table I
STAR CONDUCTANCE, CONDUCTANCE AND ARI FOR DIFFERENT CLUSTERING METHODS FOR GRAPHS WITH STAR-HEAVY CLUSTERS.

Method	Motif cond.	Conductance	ARI
Triangle CPD	7.6470	**1.1533**	0.8880
Star CPD	**6.4410**	1.1771	0.8756
Benson et al.	7.0997	1.3842	0.8032
Spectral	6.8368	1.1966	**0.8941**

Table II
SUM OF TRIANGLE AND STAR CONDUCTANCE, CONDUCTANCE AND ARI FOR DIFFERENT CLUSTERING METHODS FOR GRAPHS WITH STAR- AND TRIANGLE-HEAVY CLUSTERS.

Method	Motif cond.	Conductance	ARI
Triangle CPD	10.1179	1.6977	0.2209
Star CPD	8.7709	1.5229	0.5124
Triangle and adj. mat.	9.0428	1.4980	0.5126
Benson et al.	9.2695	1.4980	0.5700
Spectral	9.5504	1.4464	0.4999
Triangle, star and adj. mat.	**8.0264**	**1.2799**	**0.6891**

instead of stars. The spectral method achieves the highest ARI, but as the connections between the different clusters introduce new stars in the graph, the original clusters do not necessarily yield the best clustering of the graph anymore.

D. Coupled decompositions

In order to test the performance of a coupled CPD method, 100 experiments are performed on graphs with two star-heavy clusters and two triangle-heavy clusters of ten nodes (eight stars and eight triangles per cluster respectively). For each graph, 75 edges are flipped. The star and triangle conductance scores are summed in order to evaluate the overall performance for both types of clusters. The coupled CPD of the star tensor, the triangle tensor and the modified adjacency matrix is computed and its results are compared to those of other methods in Table II. We list the motif conductance, the normal conductance and the ARI for the different methods. The coupled CPD method clearly outperforms the other methods: fewer stars and triangles are cut compared to the number of these structures within the clusters for this method. The method also achieves the best results for the other two metrics. Coupled decompositions are thus useful for graphs that have clusters of mixed types.

E. Performance

The speed of the different methods is compared for graphs of increasing size. The experiments are performed on a computer with an Intel i7-6820HA CPU at 2.70GHz and 16GB of RAM. The graphs have triangle-heavy clusters consisting of ten nodes. The average results over ten runs are shown for graphs ranging from 3 to 10 clusters (i.e. 30 to 100 nodes) in Figure 12. For a graph with N clusters, $20N$ edges were flipped between the clusters. The time that is required to perform the tensorization of the graph is included in the timings. The tensor methods use sparse representations of the triangle and star tensor, as only 1 and 0.03% of their entries are non-zero, respectively. All tensor methods are significantly

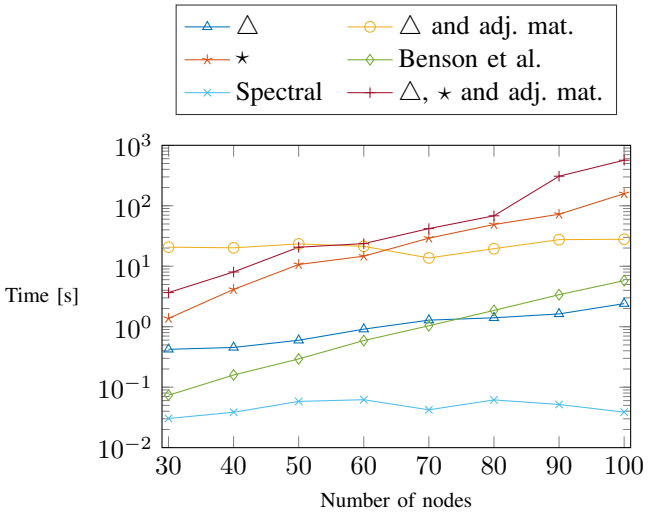

Figure 12. Timings for different clustering methods on graphs with triangle-heavy clusters. The spectral matrix method is faster than all tensor methods. The methods that use the fourth-order star tensor are clearly the slowest, followed by the method that uses a coupled CPD of the triangle tensor and the modified adjacency matrix. The tensor CPD method is about as fast as the triangle method of Benson et al.

slower than the matrix-based spectral method and their time requirements also grow faster when the number of nodes is increased. This is of course due to the higher order of the tensors. The triangle method performs on par with the method of Benson et al. Further, it can be seen that using a coupled CPD slows down the methods. Again, this is expected, as the computation of coupled decompositions is more demanding than the computation of ordinary CPDs. The fourth-order methods that use the star tensor are clearly the slowest. It is the large number of entries in the star tensor that makes the computation of the decompositions expensive.

F. Real-life example

The methods are also applied to a real-life social network: a Facebook network, graph 414 from [37], is clustered with different methods. This graph consists of 143 nodes and 1683 edges. It corresponds to a social network, so triangle and clique clustering methods are expected to perform well on this graph. Visual inspection of the graph reveals three major clusters. In Table III, the triangle conductance and conductance of the clusterings are shown for different methods. The clusterings of our triangle and coupled methods are clearly the worst. Using fourth-order clique denoising, as described above, however, the results are on par with the triangle method of Benson et al. and the spectral method. For this graph that has relatively dense clusters, using the clique tensor can thus significantly improve the results.

VII. CONCLUSION

A tensor-based framework has been introduced for clustering that exploits higher-order structures in a graph. We described how the CPD can be used to obtain a clustering from the triangle tensor of a graph. Next, we proposed alternative tensorizations that encode specific higher-order structures of

Table III
TRIANGLE CONDUCTANCE AND CONDUCTANCE OF DIFFERENT CLUSTERING METHODS FOR A REAL-LIFE FACEBOOK NETWORK.

Method	Triangle cond.	Conductance
Triangle CPD	0.3675	0.2625
Clique denoising	**0.1019**	**0.0882**
Triangle and adj. mat.	0.3603	0.2602
Benson et al.	0.1067	0.0932
Spectral	**0.1019**	**0.0882**

a graph. We also coupled multiple tensorizations of the same graph to consider different higher-order structures simultaneously. Experiments were performed on synthetic and real-life graphs to evaluate the performance of the proposed methods. The experiments confirmed that the CPD approach manages to extract graph clusters that have the desired higher-order structure. These methods can find clusterings of graphs that cut relatively few of the requested higher-order structures. Additionally, it was shown that coupled decompositions are well-suited to cluster graphs with clusters that involve more than one type of higher-order structure.

REFERENCES

[1] K. Thulasiraman and M. N. S. Swamy, *Graphs: Theory and Algorithms*. Hoboken, NJ, USA: John Wiley & Sons, 1992.

[2] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3–5, pp. 75–174, 2010.

[3] S. E. Schaeffer, "Graph clustering," *Computer Science Review*, vol. 1, pp. 27–64, 2007.

[4] S. Agarwal, K. Branson, and S. Belongie, "Higher order learning with graphs," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, June 2006, pp. 17–24.

[5] A. Prat-Pérez, D. Dominguez-Sal, J. M. Brunat, and J.-L. Larriba-Pey, "Shaping communities out of triangles," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM 2012)*, October 2012, pp. 1677–1681.

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

[7] A. Cichocki, C. Mandic, A. Phan, C. Caiafa, G. Zhou, Q. Zhao, and L. De Lathauwer, "Tensor decompositions for signal processing applications. From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, pp. 145–163, 2015.

[8] R. Henrion, "Body diagonalization of core matrices in three-way principal components analysis: Theoretical bounds and simulation," *Journal of Chemometrics*, vol. 7, no. 6, pp. 477–494, 1993.

[9] T. Schultz and G. L. Kindlmann, "Open-box spectral clustering: Applications to medical image analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2100–2108, 2013.

[10] B. W. Bader, M. W. Berry, and M. Browne, "Discussion tracking in Enron email using PARAFAC," in *Survey of Text Mining: Clustering, Classification, and Retrieval*, M. W. Berry and M. Castellanos, Eds. Springer, 2004, pp. 147–162.

[11] T. G. Kolda, B. W. Bader, and J. P. Kenny, "Higher-order web link analysis using multilinear algebra," in *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005)*, Houston, TX, USA, November 2005, pp. 242–249.

[12] E. E. Papalexakis, L. Akoglu, and D. Ienco, "Do more views of a graph help? community detection and clustering in multi-graphs," in *16th International Conference on Information Fusion (FUSION 2013)*, Istanbul, Turkey, July 2013, pp. 899–905.

[13] A. Benson, D. F. Gleich, and J. Leskovec, "Tensor spectral clustering for partitioning higher-order network structures," in *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM 2015)*, Vancouver, BC, Canada, April 2015, pp. 118–126.

[14] F. L. Hitchcock, "The expression of a tensor or a polyadic as a sum of products," *Journal of Mathematical Physics*, vol. 6, no. 1–4, pp. 164–189, 1927.

[15] R. A. Harshman, "Determination and proof of minimum uniqueness conditions for PARAFAC1," *UCLA Working Papers in Phonetics*, vol. 22, pp. 111–117, 1972.

[16] N. D. Sidiropoulos and R. Bro, "On the uniqueness of multilinear decomposition of N-way arrays," *Journal of Chemometrics*, vol. 14, pp. 229–239, 2000.

[17] J. B. Kruskal, "Three-way arrays: rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics," *Linear Algebra and its Applications*, vol. 18, no. 2, pp. 95–138, 1977.

[18] M. Sørensen and L. De Lathauwer, "New uniqueness conditions for the canonical polyadic decomposition of third-order tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 4, pp. 1381–1403, 2015.

[19] I. Domanov and L. De Lathauwer, "Canonical polyadic decomposition of third-order tensors: relaxed uniqueness conditions and algebraic algorithm," *Linear Algebra and Its Applications*, in press.

[20] ——, "On the uniqueness of the canonical polyadic decomposition of third-order tensors — Part II: Uniqueness of the overall decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 876–903, 2013.

[21] ——, "Generic uniqueness conditions for the canonical polyadic decomposition and INDSCAL," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 4, pp. 1567–1589, 2015.

[22] S. Wasserman and K. Faust, *Social network analysis*. Cambridge, UK: Cambridge University Press, 1994.

[23] M. N. Kolountzakis, G. L. Miller, R. Peng, and C. E. Tsourakakis, "Efficient triangle counting in large graphs via degree-based vertex partitioning," *Internet Mathematics*, vol. 8, no. 1-2, pp. 161–185, 2012.

[24] J. B. Kruskal, "Rank, decomposition, and uniqueness for 3-way and n-way arrays," *Multiway Data Analysis*, pp. 7–18, 1989.

[25] M. Boussé, O. Debals, and L. De Lathauwer, "A tensor-based method for large-scale blind source separation using segmentation," *IEEE Transactions On Signal Processing*, in press.

[26] D. L. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality," in *AMS Conference on Math Challenges of the 21st century (AMS 2000)*, Los Angeles, CA, USA, August 2000, pp. 1–32.

[27] N. Vervliet and L. De Lathauwer, "A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 284–295, 2016.

[28] N. Vervliet, O. Debals, L. Sorber, and L. De Lathauwer, "Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 71–79, 2014.

[29] E. E. Papalexakis, C. Faloutsos, T. M. Mitchell, P. P. Talukdar, N. D. Sidiropoulos, and B. Murphy, "Turbo-smt: Accelerating coupled sparse matrix-tensor factorizations by 200×," in *Proceedings of the 2014 SIAM International Conference on Data Mining (SDM14)*, vol. 2014, Philadelphia, PA, USA, April 2014, p. 118.

[30] E. Acar, R. Bro, and A. K. Smilde, "Data fusion in metabolomics using coupled matrix and tensor factorizations," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1602–1620, Sept 2015.

[31] M. Sørensen and L. De Lathauwer, "Coupled canonical polyadic decompositions and (coupled) decompositions in multilinear rank-$(l_{r,n}, l_{r,n}, 1)$ terms — Part I: Uniqueness," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 2, pp. 496–522, 2015.

[32] M. Sørensen, I. Domanov, and L. De Lathauwer, "Coupled Canonical Polyadic Decompositions and (Coupled) Decompositions in Multilinear Rank-$(L_{r,n}, L_{r,n}, 1)$ Terms — Part II: Algorithms," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 3, pp. 1015–1045, 2015.

[33] L. Sorber, M. Van Barel, and L. De Lathauwer, "Structured data fusion," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 586–600, 2015.

[34] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[35] R. Kannan, S. Vempala, and A. Vetta, "On clusterings: Good, bad and spectral," *Journal of the Association for Computing Machinery*, vol. 51, no. 3, pp. 497–515, May 2004.

[36] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016.

[37] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Advances in neural information processing systems*, 2012, pp. 539–547.