Multiway clustering of 3-order tensor via affinity matrix

Anonymous Author
Anonymous Institution

Abstract

We propose a new method of multiway clustering for 3-order tensors via affinity matrix (MCAM). Based on a notion of similarity between the tensor slices and the spread of information of each slice, our model builds an affinity/similarity matrix on which we apply advanced clustering methods. The combination of all clusters of the three modes delivers the desired multiway clustering. Finally, MCAM achieves competitive results compared with other known algorithms on synthetics and real datasets. Our codes are available and accessible through the anonymous link https://anonymous.4open.science/r/aistatCF24/.

1 Introduction

Tensor data is seen as multidimensional arrays that structure much complex information. This occurs in a variety of domains such as time-evolving data [Araújo et al., 2018], behavioral patterns [He et al., 2018], heterogeneous information networks [Ermiş et al., 2015], and social networks [Nickel et al., 2011]. Developing algorithms to understand the characteristics of different patterns in this data type remains a considerable challenge for data scientists. Machine learning methods define powerful statistical tools to undertake the mining of multidimensional data [Cichocki et al., 2009]. Sundry algebraic tools were developed to gather the information that lies in a subspace of the tensor dataset such as the representation of a tensor with a finite sum of rank-one decomposition [Harshman and Lundy, 1996, Kiers, 2000, Kolda and Bader, 2009]. Alternatively, one can reduce the initial tensor to a compressed tensor, i.e. to a tensor with a smaller size called **core tensor**, which leads to a different approach to understanding the dataset [Harshman and Lundy, 1996, Tucker, 1966].

One of the most prominent approaches for tensor pattern recognition declines in clustering algorithms. In particular, clustering which is unsupervised learning attracts substantial attention because of its considerable range of applications. Clustering refers to the partition of data into clusters (groups) of similar objects. Each cluster consists of objects that are similar to each other and dissimilar to objects in other groups [Berkhin, 2006].

Clustering algorithms for three-dimensional datasets are well developed nowadays. In the following, we give a lightening review of two different types of clustering methods for tensor data: some methods require the number of clusters and some others do not. The Tucker+k-means method [Huang et al., 2008] starts with the Tucker decomposition [Tucker, 1966, Kolda and Bader, 2009] and leads to the core tensor with the three membership matrices. Independent of each other, the membership matrices from the tensor decomposition keep the variation of the data in each mode. Then the k-means algorithm [Hartigan and Wong, 1979] applies to each membership matrix to determine the elements of the clusters in each mode.

The CANDECOMP/PARAFAC decomposition of a tensor [Kiers, 2000] with k-means (CP+k-means) is regarded as a particular case of the Tucker+k-means method. Indeed, in the CP decomposition, the core tensor is structured into a superdiagonal three-way tensor (for a cubical 3-order tensor, this is the body diagonal of the tensor) and the data is represented as a sum of component rank-one tensors which are the columns of the membership matrices [Kiers, 2000, Kolda and Bader, 2009]. Once again, we run a k-means to each membership matrix to determine the different clusters in a different mode.

Another key clustering method is the so-called multiway clustering via tensor block models (TBM) [Wang and Zeng, 2019]. This approach uses a particular version of the Tucker decomposition: in each mode, the corresponding membership matrix that contains only 0 and 1, becomes the clustering partition. The heterogeneous tensor decomposition for clustering via manifold optimization [Sun et al., 2016] is also based on a specific Tucker decomposition called the heterogeneous model. The clustering membership information is recorded in the

last membership matrix. This matrix gets updated over the multinomial manifold principle.

All the previous clustering methods take as a hyper-parameter the number of clusters in each mode. Other algorithms do not require such a hyper-parameter. Among those, we cite the Parameter-Less Tensor Co-clustering [Battaglia and Pensa, 2019]. This algorithm is adapted to the non-negative tensors and is based on Goodman-Kruskal's τ association measure [Goodman and Kruskal, 1979]. It maximizes the scalarization function with a stochastical local search to find the different clusters in all modes. The new input parameter needed for this algorithm is the number of iterations. We could also mention some algorithms that extract a unique cluster that contains the most relevant information satisfying some criterion of similarity. Among those, the Tensor Biclustering algorithm [Feizi et al., 2017] determines the highly correlated trajectories over the third dimension that lies in a subspace (rank-one tensor). To apply this scheme, one needs the size of the cluster in each mode as an input. More recently, using neither the number of clusters nor the cluster size, the multi-slice clustering algorithm [Andriantsiory et al., 2021] is based on the comparison of the spread of information for each tensor slice. This algorithm builds an affinity or similarity matrix between slices for each tensor mode and uses a threshold error to guarantee the similarity quality of the output cluster.

Our contribution in this paper is to present new approaches for multiway clustering for 3-order tensors based on the affinity (similarity) matrix inspired by [Andriantsiory et al., 2021]: in a given mode, each entry of the matrix represents the similarity of two slices in this mode. However, departing from the method proposed in the above reference that singles out a unique cluster, we undertake the analysis of this affinity matrix using advanced multiway-clustering algorithms based on spectral analysis [Bach and Jordan, 2004] and affinity propagation [Frey and Dueck, 2007]. These methods are particularly appropriate in our situation as they are designed for affinity matrices. We titled our method Multiway Clustering via Affinity Matrix (MCAM) where the algorithms determine the clusters in each mode independently. The combination of all clusters of the three modes provides the multiway clustering as illustrated in figure 1. Another benefit of the present method is its adaptability in two situations depending on whether the number of clusters is given as input or not. Hence, the users may choose among these two options. In addition, the MCAM is portable in the sense that one may choose any other matrix clustering algorithm to perform the clustering on its affinity matrix. In this work, as mentioned above, we focus on two well-established algorithms, spectral clustering, and affinity propagation. The MCAM performs well on both synthetic and real datasets. We compare its results with three known and well-performing clustering algorithms for tensorial data, namely Tucker+k-means, CP+k-means, and TBM. We find that the MCAM is efficient with a few numbers of the largest eigenvalues with their corresponding eigenvectors of the slice covariance matrices, whereas the other methods require a much higher number of rank-one tensors in the decomposition of the tensor dataset before becoming truly efficient. This, therefore, privileges the MCAM for generic tensor data that do not assume any particular form of the tensor data.

We present two variant algorithms of the MCAM. Both algorithms are competitive and efficient compared to other methods. The first algorithm called MCAM-I has a domain of validity for any generic tensor dataset. The second algorithm, which we call MCAM-II, works and performs better than MCAM-I in some particular situations where no correlation occurs between leading and subleading eigenvalue-eigenvector pairs of different slice covariance matrices. We find it useful to report both results for both algorithms as this already allows us to gauge the clustering efficiency of MCAM.

The structure of this paper is the following. Section 2 and section 3 present respectively the methodology of MCAM and derive two algorithms. For the reproducibility purpose, we open-sourced our experimental setup and model implementations at the anonymous link https://anonymous.4open.science/r/anonym_mcam/. Section 4 discusses some experiments on synthetic and real datasets. With the results, we conclude that the MCAM is a valid multiway cluster detector. We also compare the performance of MCAM with three known algorithms (CP+k-means, Tucker+k-means and TBM) before summarizing our results in a conclusion in section 5.
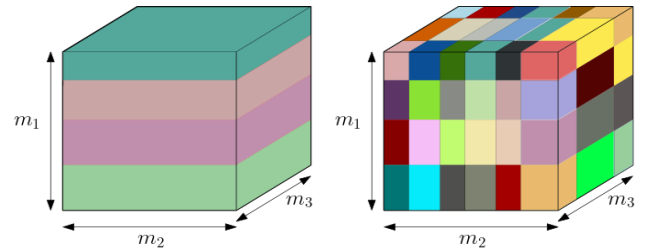

Figure 1: Mode-1 clustering (left) and multiway clustering (right) of a 3-order tensor.

2 Methodology

2.1 Notation

We denote by $\mathcal{T}, \mathcal{X}, \mathcal{Z}$ the tensor dataset, the signal tensor and the noise tensor, respectively. Matrices are represented by capital letters $(T, X, Z, \cdots)$. For a matrix V, $|V|$ denotes a matrix with the same size as V and its en-

tries are the absolute values of the corresponding entries in V. V^t represents the transpose of the matrix V. Vectors and real numbers are denoted by boldface lowercase letters and lowercase letters, respectively. For a nonzero integer n, $[n] = \{1, 2, \cdots, n\}$. For a matrix, the largest eigenvalue is called the **top eigenvalue**, and its corresponding eigenvector **top eigenvector**.

We consider a tensor $\mathcal{T} = \mathcal{X} + \mathcal{Z}$ with $\mathcal{T} \in R^{m_1 \times m_2 \times m_3}$, where $\mathcal{X}$ is the signal tensor, and $\mathcal{Z}$ is the noise tensor. The CP decomposition [Kolda and Bader, 2009] consists in writing the tensor in the form of a finite sum of the rank-one tensors. It can be viewed as a Tucker decomposition where the core tensor is superdiagonal. Hence, for a signal tensor $\mathcal{X}$, its CP decomposition with d rank-one tensors (d a positive integer) finds the form

$$\mathcal{X} \approx \sum_{j=1}^{d} \gamma_j \mathbf{u}_j \otimes \mathbf{v}_j \otimes \mathbf{w}_j, \qquad (1)$$

then $\mathcal{T}$ can be written as

$$\mathcal{T} \approx \sum_{j=1}^{d} \gamma_j \mathbf{u}_j \otimes \mathbf{v}_j \otimes \mathbf{w}_j + \mathcal{Z}, \qquad (2)$$

where $\mathbf{u}_j \in R^{m_1}, \mathbf{v}_j \in R^{m_2}, \mathbf{w}_j \in R^{m_3}, \|\mathbf{u}_j\| = \|\mathbf{v}_j\| = \|\mathbf{w}_j\| = 1$, for $\gamma_j \in R_+^*$ for all $j \in \{1, \cdots, d\}$ and $d \ll \min(m_1, m_2, m_3)$, γ_j is the weight of each rank-one component of the tensor.

We assume that the entries of the noise tensor $\mathcal{Z}$ are independent identically distributed (i.i.d) standard normal random variables.

Given a mode, the matrix that records the slice similarity is called affinity or similarity matrix. The idea of constructing the matrix representing the slice similarity in each mode of the tensor has been introduced in [Andriantsiory et al., 2021] and has led to the so-called the multi-slice clustering for a rank-one 3-order tensor. This algorithm delivers a cluster of slices based on a threshold parameter that gauges the similarity between slices. We bring an improvement to this method to determine the multiway clustering of 3-order tensor. Note that, finally, there is no straightforward comparison between the resulting clustering algorithms of the two methods: the multi-slice clustering delivers a single cluster of slices in each mode of the tensor and no guarantee of the similarity of the remaining dataset, whereas MCAM delivers a partition of the slices in each mode, from which we identify several clusters with guaranteed similarity.

The strategy of the MCAM is declined in two phases. Firstly, we construct the similarity matrix for a rank-one tensor dataset. We generalize this construction of the similarity matrix for any tensor dataset. Secondly, once the similarity matrix is obtained, the next task is to apply an advanced matrix clustering to deliver the multiway clustering. We will focus on the so-called spectral clustering method [Bach and Jordan, 2004] and the affinity propagation [Frey and Dueck, 2007] to extract the set of clusters in each mode. Note that the following explanations concentrate on the mode-1 of the tensor and there is no issue with extending the reasoning for the remaining modes.

2.2 Affinity matrix for a rank-one tensor (equation (2) with $d = 1$)

We use the Matlab notation for the manipulation of the tensor. In mode-1 of the tensor $\mathcal{T}$, we have m_1 slices, and the i-th slice is defined by:

$$T_i = \mathcal{T}(i, :, :) = \mathcal{X}(i, :, :) + \mathcal{Z}(i, :, :), \qquad \forall i \in [m_1] \quad (3)$$

where $\mathcal{X}(i, :, :)$ and $\mathcal{Z}(i, :, :)$ are respectively the i-th slice of the signal tensor and the i-th slice of the noise tensor, and T_i is a matrix in $R^{m_2 \times m_3}$. The principal component of the columns of T_i is the largest eigenvector of $T_i^t T_i$ [Shlens, 2014]. The covariance matrix of T_i is expressed as the following:

$$T_i^t T_i = \hat{\gamma}_1^2 \mathbf{w}_1 \mathbf{w}_1^t + W_i, \qquad (4)$$

where $\hat{\gamma}_1 = \gamma_1 \mathbf{u}_1(i)$,
and $W_i = X_i^t Z_i + Z_i^t X_i + Z_i^t Z_i$.

The first term of equation (4) represents the covariance matrix of the signal slice X_i and W_i refers to the noise. The mismatch between the subspace of the spectral decomposition of the slice $T_i^t T_i$ and the signal slice $X_i^t X_i$ is bounded (see Davis-Kahan $\sin \theta$ theorem [Yu et al., 2014]). For this noise model, the relationship between the top eigenvector of the covariance matrix of the i-th slice and the corresponding signal is also controlled by the following proposition (see lemma 2, supplementary material of [Andriantsiory et al., 2021]).

With the standard Gaussian noise model, let $\hat{\mathbf{w}}_1$ be the top eigenvector of $T_i^t T_i \in R^{m_3 \times m_3}$, and $\hat{\gamma}_1 = \mathcal{O}(m_3)$, and $\alpha = \|\mathbf{w}_1\|_\infty$. We have

$$\|\hat{\mathbf{w}}_1 - \mathbf{w}_1\|_\infty \le \mathcal{O}\left(\frac{1}{\hat{\gamma}_1} \alpha \log(m_3)\right) \qquad (5)$$

with high probability as $m_3 \to \infty$.

The equation (5) indicates that, for a large m_3, if the signal weight becomes large then the two vectors $\hat{\mathbf{w}}_i$ and $\mathbf{w}_i$ become more similar.

Each slice is represented by its top eigenvalue and top eigenvector. These provide the direction in which the data has maximum variance and in which the data is most spread out. If one deals with a noise slice, we can determine the variance or top eigenvalue associated with its top eigenvector. An estimate of such a quantity is possible as the

covariance matrix of a random standard Gaussian matrix, it has a white Wishart distribution [Johnstone, 2001]. We can therefore approximate the top eigenvalue distribution of the covariance matrix by the Tracy-Widom distribution [Chiani, 2014].

In mode-1, let's consider V in $R^{m_3 \times m_1}$ a matrix with the column vector $\mathbf{x}_i = (\lambda_i/\lambda) \times \hat{\mathbf{w}}_i$, where $\lambda = \max_{i \in [m_1]} \lambda_i$, λ_i is the top eigenvalue and $\hat{\mathbf{w}}_i$ is the top eigenvector of the i-th slice covariance matrix $T_i^t T_i$ for $i \in [m_1]$. From now on, we drop the hat from the notation of $\hat{\mathbf{w}}_i$, as there will be no possible confusion.

The similarity matrix, called C', of all slices associated with the mode-1 is defined as:

$$C' = |V^t V| \tag{6}$$

where the similarity of the slice i and slice j

$$C'_{ij} = \text{sim}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_i \sigma_j |\langle \mathbf{w}_i, \mathbf{w}_j \rangle|$$

and where $0 \leq \text{sim}(\mathbf{x}_i, \mathbf{x}_j) \leq 1$ and $\sigma_i = \lambda_i/\lambda$ for all i.

The slices i and j are similar if C'_{ij} is close to 1, and they are dissimilar if C'_{ij} is close to zero.

2.3 Affinity matrix for general 3-order tensor

We assume that our tensor dataset is approximated by a signal of d rank-one tensors plus a noise term. Building the similarity matrix by only considering the top eigenvalue and top eigenvector of each slice may result in a loss of information for each slice because the exact rank-one decomposition of the tensor is unknown most of the time. To solve this issue, we propose a new construction method of the similarity matrix C' that takes into account multiple eigenvalues and eigenvectors for each slice.

To illustrate the new method, namely the MCAM, we assume that each slice is represented by their r largest eigenvalues and their corresponding eigenvectors, $r \leq d$. For simplicity, let's limit to the case of two slices. The generic case can be inferred easily from this point. The ideal situation for two slices to be similar is that the r eigenvalues and eigenvectors of the two matrices are pairwise identical. Without loss of generality, we take $r = 2$ for the following explanation and the computation of the similarity matrix. Once more, the case $r \geq 2$ can be recovered without difficulties as it involves all possible eigen-subspace pairs.

For fixed i-th and j-th slices, we select the eigenspace $(\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)})$ and $(\mathbf{x}_1^{(j)}, \mathbf{x}_2^{(j)})$, respectively, such that

$$\mathbf{x}_a^{(i)} = (\lambda_a^{(i)}/\lambda)\mathbf{w}_a^{(i)}, \quad a \in \{1, 2\} \tag{7}$$

where λ keeps its meaning as $\max_{i \in [m_1]} \lambda_1^{(i)}$.

By definition of the eigenspace, we have $\lambda_1^{(i)} \geq \lambda_2^{(i)}$ and $\lambda_1^{(j)} \geq \lambda_2^{(j)}$. The question here is: given a strong

(resp. weak) correlation $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_1^{(j)} \rangle| \sim 1$ (resp. ~ 0), what does this entail on the cross similarities $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ and $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_1^{(j)} \rangle|$? Note that there are two ways to obtain $|\langle \mathbf{x}, \mathbf{x}' \rangle| \sim 0$: either $\mathbf{x} \perp \mathbf{x}'$ or the norm of one of these vectors gets close to 0. To answer to the above question, we conduct the following case study:

- Assume that $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_1^{(j)} \rangle|$ is close to one, then the cross similarities $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ and $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_1^{(j)} \rangle|$ are close to zero due to the orthogonality of the vectors $\mathbf{x}_1^{(i)} \perp \mathbf{x}_2^{(i)}$ and $\mathbf{x}_1^{(j)} \perp \mathbf{x}_2^{(j)}$.

- Assume that $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_1^{(j)} \rangle|$ is close to zero. In this case, two possibilities may occur:

  - If $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ close to one, then the cross similarities $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ and $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_1^{(j)} \rangle|$ become close to zero because of the orthogonality of the vectors in the slice eigen-subspace.
  - If $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ close to zero, then we need to push further the analysis. We have the following table, where the choice of $i, j \in \{1, 2\}$ holds without loss of generality:

	$\mathbf{x}_1^{(i)} \perp \mathbf{x}_1^{(j)}$	$\lambda_1^{(i)}/\lambda \sim 0$
$\mathbf{x}_2^{(i)} \perp \mathbf{x}_2^{(j)}$	?	$\checkmark$
$\lambda_2^{(i)}/\lambda \sim 0$	?	$\checkmark$

(8)

where the symbol $\checkmark$ indicates the cases where the cross similarities $|\langle \mathbf{x}_1^{(i)}, \mathbf{x}_2^{(j)} \rangle|$ and $|\langle \mathbf{x}_2^{(i)}, \mathbf{x}_1^{(j)} \rangle|$ become close to 0. In the other cases, no conclusion can be reached.

The above case analysis reveals that, in some situations, the cross scalar products $|\langle \mathbf{x}_k^{(i)}, \mathbf{x}_{k'}^{(j)} \rangle|$ may become close to 1 and can be non-negligible. In general, they need to be incorporated in the analysis of the affinity matrix. However, in the following and some cases, the experiments show that their presence may result in less good clustering performance. This deserves better understanding and will be left for future research. Taking this into account, we implement two variant algorithms. Regarded valid to a full extent, one algorithm (MCAM-I) includes the cross scalar products in the affinity matrix. The second algorithm (MCAM-II) neglects these cross terms. We will perform our analysis in parallel keeping track of the performance of both algorithms.

For the algorithm MCAM-I, the matrix is a linear combination of matrices $C_{kk'}$ for $k, k' = 1, \ldots, r$, with entry $(C_{kk'})_{ij} = |\langle \mathbf{x}_k^{(i)}, \mathbf{x}_{k'}^{(j)} \rangle|$. Then, we obtain r^2 matrices $(C_{11}, C_{12}, \cdots, C_{rr})$ from the eigen-subspaces made of the r top eigenvectors of each slice. All matrices $(C_{kk'})_{k,k' \in [r]}$

are combined to build the similarity matrix C' defined as:

$$C' = \frac{\lambda^2}{(\sum_{k=1}^r \lambda_k)^2} \sum_{k=1}^r \sum_{k'=1}^r C_{kk'} \qquad (9)$$

On the other hand, MCAM-II proposes to take the linear combinaison of the matrices $C_k := C_{kk}$ for $k = 1, \cdots, r$. Then we obtain r matrices $(C_{11}, C_{22}, \cdots, C_{rr})$ from the eigen-subspaces made of r top eigenvectors of each slice. The similarity matrix C' defined as :

$$C' = \frac{\lambda^2}{\sum_{k=1}^r \lambda_k^2} \sum_{k=1}^r C_{kk}, \qquad (10)$$

In both methods, $\lambda_k = \max_i \lambda_k^{(i)}$ (in this sense $\lambda = \lambda_1$ of equation (7)).

If the entry C'_{ij} of the matrix C' is close to one, this means that the i-th and j-th slices are similar. Otherwise, C'_{ij} close to zero indicates that the two slices are dissimilar.

We realize that the computational cost of the construction of similarity matrix of MCAM-I dominates that of the matrix of MCAM-II. However, our experiments show that r is generally quite a low integer, and thus the overall computational complexity of both algorithms is equivalent.

One main issue in the above formalism is the determination of (the best estimation of) r as this is indeed an unknown from a generic tensor data input. This is the same as determining the best number of principal components (PC) [Shlens, 2014] for a given matrix. Different methods are proposed to determine the number of PC for the best reconstruction of the data. Among these, we mention the scree plot [Cattell, 1966] and the ratio methods [Abdi and Williams, 2010]. The scree plot strategy displays the eigenvalues sorted in decreasing order and a threshold selection parameter for the vertical (eigenvalue) axis. Then, it keeps the components with values above the threshold and removes the remaining. A good threshold is determined at the point where the eigenvalues drop significantly.

For a fixed slice i, the ratio method uses $\lambda_j^{(i)} / \sum_k \lambda_k^{(i)}$ where $\lambda_j^{(i)}$ is the eigenvalue related to the j-th PC. Here, one sorts the ratio $(\lambda_j^{(i)} / \sum_k \lambda_k^{(i)})_j$ in decreasing order and selects the indices up to a significant drop. For our problem, it turns out that both ideas are equivalent.

The following approach fixes the value of r. We have m_1 slices and the covariance matrix of each slice has m_3 eigenvectors. For the slice i, we denote by n_i the number of eigenvectors selected by scree plot method (equivalently by the ratio method) and choose $r = \max_i n_i$. The implications of such a choice is as follows: consider a j-th slice that has a number of selected vectors less than r, i.e. $n_j < r$. Because the significant drop of the eigenvalues of the covariance matrix of the j-th slice happens at the posi-

tion n_j, the remaining $r - n_j > 0$ eigenvectors are meaningless for j-th slice. Thus, adding them will not affect the clustering process.

2.4 Cluster selection

For each mode, we have the similarity matrix C' which records the similarity between all slices of this mode. To perform the clustering from these similarity matrices, we use the spectral clustering (SC) [Bach and Jordan, 2004] if the desired number of clusters in the three modes is given as an input ($k = (k_1, k_2, k_3)$), otherwise, we use the affinity propagation algorithm (AP) [Frey and Dueck, 2007] that does not need such data.

3 Algorithms

In this section, we present the two algorithms of MCAM for $3-$order tensors. As indicated in the previous section, these algorithms divide into two steps. The first step constructs the similarity matrix C' using the expression (9) in one case, and (10) in the other. The second step uses the matrix C' and applies a given clustering scheme (AP or SC) according to the user choice. The first option only needs the tensor data set as an input: we use the AP algorithm to detect the clusters. The algorithm 1 [1] allows this option. A second option requests the tensor data set and the desired number of clusters of each mode as an input. In this case, we use the SC algorithm to find the different clusters. The two algorithms easily adapt in such a situation. The output of the algorithm $(\mathcal{C}_i)_{i \in [3]}$ represents respectively the clusters inside the three modes.

We present also a second version of the algorithm, that neglects cross similarities (algorithm 2). Therein, the lines 9 to 12 replace the lines 9 to 15 of the algorithm 1.

Computational complexity To simplify the evaluation, we request $m_i \in \Theta(n)$ for $i = 1, 2, 3$, i.e. we fix $m_1 = n$ and other dimensions are comparable with n. The construction of the covariance matrix of each slice has a complexity $\mathcal{O}(n^3)$ and the spectral decomposition has at most $\mathcal{O}(n^3)$. Then, the computation of the covariance through all slices costs $\mathcal{O}(n^4)$ complexity. SC has a complexity $\mathcal{O}(n^3)$ and AP has a complexity at most $\mathcal{O}(n^4)$. So, the MCAM algorithm has a complexity class $\mathcal{O}(n^4)$.

4 Experimentation

In the following experiments, we apply the MCAM algorithms to synthetical datasets and one real dataset. For the synthetical datasets, the number of clusters in each mode and the ground truth of elements are known. Hence,

---

[1]The code is available at: https://anonymous.4open.science/r/aistatCF24/

Algorithm 1 Multiway clustering via affinity matrix (MCAM-I)

1: **Input:** 3-order tensor $\mathcal{T} \in R^{m_1 \times m_2 \times m_3}$.
2: **for** j in $\{1, 2, 3\}$ **do**
3:    **for** i in $\{1, 2, \cdots, m_j\}$ **do**
4:       Compute the spectral decomposition of $T_i^t T_i$.
5:       Store the normalized eigenvectors
        $(\mathbf{x}_1^{(i)}, \cdots, \mathbf{x}_k^{(i)}, \cdots)$
6:       Compute n_i (scree-plot method).
7:    **end for**
8:    $r \leftarrow \max_i n_i$
9:    **for** k in $\{1, 2, \cdots, r\}$ **do**
10:       $V_k \leftarrow [\mathbf{x}_k^{(1)} \, \mathbf{x}_k^{(2)} \cdots \mathbf{x}_k^{(m_j)}]$
11:       $C_{kk} \leftarrow |V_k^t V_k|$
12:       **for** k' in $\{1, 2, \cdots, r\} \setminus \{k\}$ **do**
13:          $C_{kk'} \leftarrow |V_k^t V_{k'}|$
14:       **end for**
15:    **end for**
16:    $C' \leftarrow \frac{\lambda^2}{(\sum_{k=1}^r \lambda_k)^2} \sum_{k=1}^r \sum_{k'=1}^r C_{kk'}$.
17:    $\mathcal{C}_j \leftarrow$ Affinity Propagation(C').
18: **end for**
19: **Output:** $(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3)$, the elements of the different clusters of the three modes.

Algorithm 2 Multiway clustering via affinity matrix (MCAM-II)

1: **Input:** 3-order tensor $\mathcal{T} \in R^{m_1 \times m_2 \times m_3}$.
2: **for** j in $\{1, 2, 3\}$ **do**
3:    **for** i in $\{1, 2, \cdots, m_j\}$ **do**
4:       Compute the spectral decomposition of $T_i^t T_i$.
5:       Store the normalized eigenvectors
        $(\mathbf{x}_1^{(i)}, \cdots, \mathbf{x}_k^{(i)}, \cdots)$
6:       Compute n_i (scree-plot method).
7:    **end for**
8:    $r \leftarrow \max_i n_i$
9:    **for** k in $\{1, 2, \cdots, r\}$ **do**
10:       $V_k \leftarrow [\mathbf{x}_k^{(1)} \, \mathbf{x}_k^{(2)} \cdots \mathbf{x}_k^{(m_j)}]$
11:       $C_k \leftarrow |V_k^t V_k|$
12:    **end for**
13:    $C' \leftarrow \frac{\lambda^2}{\sum_{k'} \lambda_{k'}^2} \sum_{k=1}^r C_{kk}$.
14:    $\mathcal{C}_j \leftarrow$ Affinity Propagation(C').
15: **end for**
16: **Output:** $(\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3)$, the elements of the different clusters of the three modes.

---

we evaluate the quality of the MCAM output by comparing it with the true cluster. We also compare the performance of our algorithms with three known clustering algorithms for multidimensional datasets: the CP+k-means, the Tucker+k-means and the multiway clustering via tensor block models (TBM). For the real data set, we evaluate the clustering quality by computing the root means square error (RMSE) between the initial data and the estimated tensor generated from the clustering result.

4.1 Synthetical datasets

We compare the output of MCAM to the other algorithms with a synthetic data. To do so, we generate the tensor datasets with the CP-decomposition. We thus evaluate the clustering quality of the output of each algorithm by computing their clustering quality criteria: the Adjusted Random Index (ARI) [Hubert and Arabie, 1985] and the Normalized Mutual Information (NMI) [Strehl and Ghosh, 2003]. In all situations, we show that MCAM performs with compelling results.

We generate a 3-order tensor dataset $\mathcal{T} \in R^{m_1 \times m_2 \times m_3}$ as defined in the equation (2),

$$
\begin{aligned}
\mathcal{T} &= \sum_{j=1}^d \gamma_j \mathbf{u}_j \otimes \mathbf{v}_j \otimes \mathbf{w}_j + \mathcal{Z} \qquad (11) \\
&= [\Lambda; A, B, C] + \mathcal{Z}
\end{aligned}
$$

where $\Lambda = (\gamma_1, \gamma_2, \cdots, \gamma_d)$, $A = [\mathbf{u}_1 \cdots \mathbf{u}_d]$, $B = [\mathbf{v}_1 \cdots \mathbf{v}_d]$ and $C = [\mathbf{w}_1 \cdots \mathbf{w}_d]$.

The columns of the matrices A are orthogonal, and so are those of B and C [Kolda and Bader, 2009]. γ_j is a positive weight of the j-th rank-one tensor of the dataset, for all $j \in [d]$. $\mathcal{Z}$ is a Gaussian noise tensor as defined in equation (2). We generate a tensor of size $m_1 = m_2 = m_3 = 100$ and each mode has the same number of clusters $c = 9$. We fix the value of $(\gamma_j)_{j \in [d]}$, by assuming that $\gamma_1 = \gamma_2 = \cdots = \gamma_d = \gamma$. In each mode, we denote by $(J_i)_i$ the clusters. For each mode and for $j \in \{1, \cdots, d\}$, the column of A, B and C defined as $\mathbf{u}_j(i) = 1/\sqrt{|J_j|}$, $\mathbf{v}_j(i) = 1/\sqrt{|J_j|}$ and $\mathbf{w}_j(i) = 1/\sqrt{|J_j|}$ for $i \in J_j$ otherwise, $\mathbf{u}_j(i) = \mathbf{v}_j(i) = \mathbf{w}_j(i) = 0$. We vary the value of γ from 30 to 80. For each value of γ, we repeat the experiment 10 times and present the mean and the standard deviation of the clustering quality index (ARI). For each experiment, we regenerate the dataset and run the MCAM (I and II) with SP (coined in the figure MCAM-I/II-SP) or the MCAM (I and II) with AP (with index MCAM-I/II-AP) to the matrix C'. We also exhibit the performance of CP+k-means with rank decomposition d equals the number of the clusters (i.e $d = c$), the performance of Tucker+k-means with rank decomposition equals to (c, c, c) because

we have c clusters in each mode, as well as the performance of the TBM. For the convergence of the TBM, we use 50 iterations to update the membership matrices and the core tensor. Figure 2 illustrates the performances of the different algorithms.
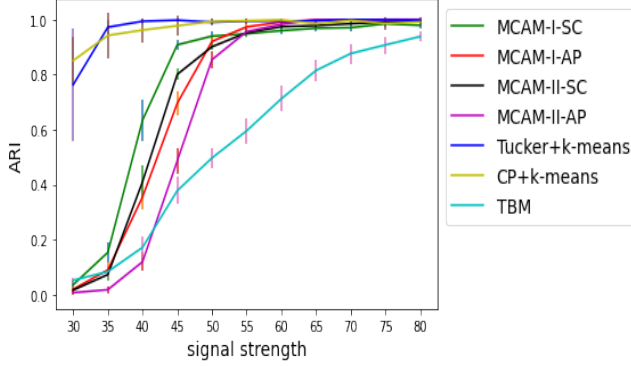

Figure 2: Comparison of the effectiveness between MCAM and the other multiway clustering algorithms with a n rank-one tensor dataset ($n = 9$).

Figure 2 shows that the MCAM-AP (I and II) performs well when the value of γ increases. We have a signal tensor with 9 clusters and they cover more than 95% of the entries of the tensor dataset. Naturally, this improves the clustering results of the MCAM-AP method. Furthermore, for the present data, we see that MCAM-I performs better than MCAM-II either with SC or with AP tools.

The graphics also reveal that for a small value of γ CP+k-means and Tucker+k-means have the best performance. Remember that the rank decomposition in these methods is equal to the number of clusters in each mode, and this favors them versus the other methods. (The following paragraph discusses this fact). From $\gamma = 55$, the MCAM has the quality of the CP+k-means and Tucker+k-means methods. However, good results from the TBM require a higher value of γ (more than 80) to detect all the expected clusters in this dataset model.

For $\gamma \geq 55$, MCAM-AP (I and II) recovers the expected number of clusters inside the data, i.e. $c = 9$ with their respective elements, without taking the number of clusters as an input, as opposed to the other algorithms. By performing 10 times the experiments, for both MCAM-I and -II methods, the most frequent value of r is determined at 2. This means that the MCAM algorithms only needs the two largest eigenvectors of each slice to build the similarity matrix (equation (9)) and cluster the data.

Another aspect needs emphasis: the CP+k-means or Tucker+k-means perform a good clustering only with a high enough d of rank-one decomposition (see figure 3). Thus, the higher d is, the better CP+k-means or Tucker+k-means will behave. On the other hand, the MCAM does not

require this rank decomposition d but generates the number r, which we will call "effective clustering dimension" (ECD). It is an interesting question to ask whether the access of d rank-one decomposition has a greater cost than that of the ECD r (just linear in the dimension of slices), in which case, MCAM would be even more efficient than the other algorithms. We observe that even at $r \simeq 2$ (for $\gamma \geq 55$), MCAM delivers qualitatively excellent results.
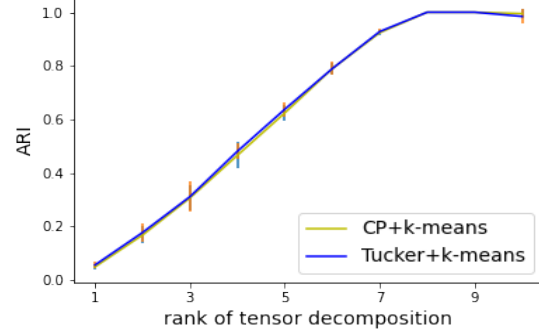

Figure 3: The two figures represent the quality of the clustering output according to the rank of the tensor decomposition, CP decomposition and Tucker decomposition, with $\gamma = 55$.

To study the dependence of the MCAM algorithm on r, we run the algorithms 1 and 2 by varying the value of r from 1 to 10 (see equations (9) and (10)) by constructing the similarity matrix associated with the dataset with signal strength $\gamma = 55$. For each r, we run the algorithm MCAM-SC 10 times. Therefore, we have 10 quality indices of the output (ARI) and we compute their standard deviation and their mean. The figure 4 show the 10 means of the quality indices of the output from the two different algorithms of MCAM. We observe that the values are very close to 1. Hence, even at $r = 1$, the MCAM can provide a good output (ARI ≥ 0.95 for MCAM-I and MCAM-II). This preliminary analysis suggests that MCAM can be efficient up to a fixed and small integer r.

4.2 Real Data

Let us address the case of the real dataset. We run the MCAM-I and -II on the flow injection analysis (FIA) dataset [Norgaard and Ridder, 1994]. This dataset has a size of 12 (samples) $\times 100$ (wavelengths) $\times 89$ (times). For each mode of this tensor data, we do not have any information about the exact number of the clusters. For MCAM-SC, CP+k-means, Tucker+k-means and TBM, we apply the silhouette score [Rousseeuw, 1987] to the similarity matrix C' to detect the number of clusters in each mode. We evaluate a range of the possible number of clusters c in each mode, (the result is plotted in figure 5.For the CP+k-means algorithm, the number of rank-one decompositions
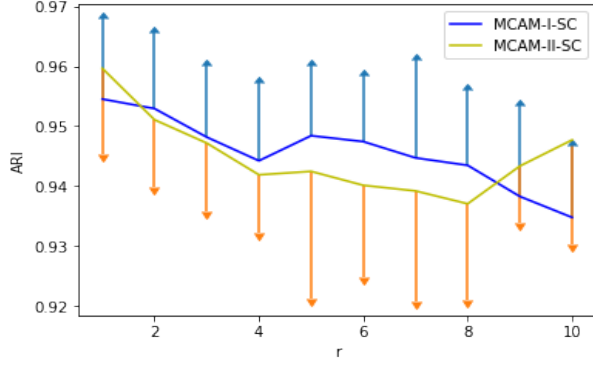
Figure 4: The clustering quality (ARI) for r varying from 1 to 10 and $\gamma = 55$.

of the tensor is equal to the maximum among the number of the clusters in the three modes ($d = \max(c_1, c_2, c_3)$). For the Tucker+k-means algorithm, the rank decompositions of the tensor is equal to the number of the clusters in each mode (rank $= (c_1, c_2, c_3)$). The RMSE [Chai and Draxler, 2014] will serve as an index of comparison of performance and quality between the different algorithms.
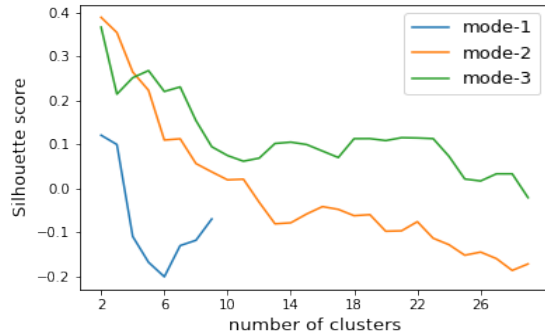

Figure 5: The silhouette score of the mode-1, of the mode-2, and of the mode-3.

For mode-1, we vary the number of clusters c from 2 to 6. For mode-2 and mode-3, we variate the number of clusters from 2 to 30. Using the silhouette score, we infer that there are 2 clusters in each mode and therefore we have $2 \times 2 \times 2 = 8$ sub-tensors in the dataset (see figure 5). The quality of the output of the algorithms is shown in table 1. Once again, the MCAM outperforms its two rivals methods CP+k-means and Tucker+k-means by having a smaller RMSE. We assume to reach the convergence of TBM after 50 iterations. The RMSE tells us that TBM has the best clustering result.

It is noteworthy that the MCAM-AP (I and II) delivers directly a different number of clusters for each mode as $(4, 6, 9)$. Hence, it is expected that it has a low RMSE

Table 1: The mean of RMSE of all sub-tensors for the number of clusters selected by Silhouette method.

METHOD	RMSE
MCAM-I-AP	0.0685
MCAM-I-SC	0.0831
MCAM-II-AP	0.0685
MCAM-II-SC	0.0831
CP+K-MEANS	0.0843
TUCKER+K-MEANS	0.0842
TBM	0.0661

mean compared to the other methods. At this moment, We see that the TBM has the best clustering result among the four algorithms which take the number of clusters as an input. We also realize that MCAM provide a better clustering than produced by the CP+k-means and Tucker+k-means.

5 Conclusion

The multiway clustering aims at partitioning all entries of a tensor into pairwise disjoint sub-tensors that define the clusters. The MCAM introduced in this paper is a new multiway clustering algorithm based on affinity matrices that record the data similarity between the tensor slices. In our procedure, the cluster selection in each mode proceeds independently from the other modes. For each mode, the determination of the cluster is divided into two parts: the first part is the construction of the similarity matrix C' and the second part is the selection of the elements of each cluster. We have proposed two algorithm versions (MCAM-I and MCAM-II), both using two matrix clustering subroutines (SC and AP). Based on the affinity propagation method, one subroutine (AP) makes the method more generic than several clustering algorithms (MCAM-AP), as it does not necessarily take the number of clusters as an input. The second method MCAM-SC (for versions -I and -II) uses spectral clustering but requires the number of clusters as an input. The MCAM procedure is evaluated on a standard synthetic and one real dataset providing excellent results. The clustering obtained from the MCAM is compared to the upshots of three other algorithms (CP+k-means, Tucker+k-means and TBM). In all experiments with a synthetic dataset, the MCAM shows a high performance compared to the other methods. Finally, the strong point of MCAM is that the clustering selection proceeds via an affinity matrix to which other methods might apply. Translating the tensor data into matrix data could have a great advantage on the scalability of the algorithm. This deserves further investigation. Another extension of this work should deepen the understanding of the similarity matrix of the MCAM method.

References

[Abdi and Williams, 2010] Abdi, H. and Williams, L. J. (2010). Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

[Andriantsiory et al., 2021] Andriantsiory, D. F., Geloun, J. B., and Lebbah, M. (2021). Multi-slice clustering for 3-order tensor. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 173–178. IEEE.

[Araújo et al., 2018] Araújo, M., Ribeiro, P., and Faloutsos, C. (2018). Tensorcast: forecasting time-evolving networks with contextual information. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5199–5203.

[Bach and Jordan, 2004] Bach, F. and Jordan, M. (2004). Learning spectral clustering. *Advances in neural information processing systems*, 16(2):305–312.

[Battaglia and Pensa, 2019] Battaglia, E. and Pensa, R. G. (2019). Parameter-less tensor co-clustering. In *International Conference on Discovery Science*, pages 205–219. Springer.

[Berkhin, 2006] Berkhin, P. (2006). A survey of clustering data mining techniques. In *Grouping multidimensional data*, pages 25–71. Springer.

[Cattell, 1966] Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate behavioral research*, 1(2):245–276.

[Chai and Draxler, 2014] Chai, T. and Draxler, R. R. (2014). Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250.

[Chiani, 2014] Chiani, M. (2014). Distribution of the largest eigenvalue for real wishart and gaussian random matrices and a simple approximation for the tracy–widom distribution. *Journal of Multivariate Analysis*, 129:69–81.

[Cichocki et al., 2009] Cichocki, A., Zdunek, R., Phan, A. H., and Amari, S.-i. (2009). *Nonnegative matrix and tensor factorizations: applications to exploratory multiway data analysis and blind source separation*. John Wiley & Sons.

[Ermiş et al., 2015] Ermiş, B., Acar, E., and Cemgil, A. T. (2015). Link prediction in heterogeneous data via generalized coupled tensor factorization. *Data Mining and Knowledge Discovery*, 29(1):203–236.

[Feizi et al., 2017] Feizi, S., Javadi, H., and Tse, D. (2017). Tensor biclustering. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 1311–1320.

[Frey and Dueck, 2007] Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points. *science*, 315(5814):972–976.

[Goodman and Kruskal, 1979] Goodman, L. A. and Kruskal, W. H. (1979). Measures of association for cross classifications. *Measures of association for cross classifications*, pages 2–34.

[Harshman and Lundy, 1996] Harshman, R. A. and Lundy, M. E. (1996). Uniqueness proof for a family of models sharing features of tucker's three-mode factor analysis and parafac/candecomp. *Psychometrika*, 61(1):133 – 154.

[Hartigan and Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

[He et al., 2018] He, J., Li, X., Liao, L., and Wang, M. (2018). Inferring continuous latent preference on transition intervals for next point-of-interest recommendation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 741–756. Springer.

[Huang et al., 2008] Huang, H., Ding, C. H. Q., Luo, D., and Li, T. (2008). Simultaneous tensor subspace selection and clustering: the equivalence of high order svd and k-means clustering. pages 327–335.

[Hubert and Arabie, 1985] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2(1):193–218.

[Johnstone, 2001] Johnstone, I. M. (2001). On the distribution of the largest eigenvalue in principal components analysis. *Annals of Statistics*, 29(2):295–327.

[Kiers, 2000] Kiers, H. A. (2000). Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 14(3):105–122.

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

[Nickel et al., 2011] Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, page 809–816.

[Norgaard and Ridder, 1994] Norgaard and Ridder, C. (1994). Rank annihilation factor analysis applied to flow injection analysis with photodiode-array detection. Chemometrics and Intelligent Laboratory, Systems 23:107 http://www.models.life.ku.dk/Flow$_Injection$.

[Rousseeuw, 1987] Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

[Shlens, 2014] Shlens, J. (2014). A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*.

[Strehl and Ghosh, 2003] Strehl, A. and Ghosh, J. (2003). Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(null):583–617.

[Sun et al., 2016] Sun, Y., Gao, J., Hong, X., Mishra, B., and Yin, B. (2016). Heterogeneous tensor decomposition for clustering via manifold optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(3):476–489.

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

[Wang and Zeng, 2019] Wang, M. and Zeng, Y. (2019). Multiway clustering via tensor block models. *Advances in neural information processing systems*, 32.

[Yu et al., 2014] Yu, Y., Wang, T., and Samworth, R. J. (2014). A useful variant of the Davis–Kahan theorem for statisticians. *Biometrika*, 102(2):315–323.