Coupled Tensor Decomposition: a Step Towards Robust Components

Matthieu Genicot*, P.-A. Absil*, Renaud Lambiotte†, and Saber Sami‡
*ICTEAM Institute, Université catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium
Email: matthieu.genicot@uclouvain.be
†naXys, University of Namur, B-5000 Namur, Belgium
‡Department of Clinical Neurosciences, University of Cambridge, Cambridge CB3 0SZ, UK

***Abstract*—Combining information present in multiple datasets is one of the key challenges to fully benefit from the increasing availability of data in a variety of fields. Coupled tensor factorization aims to address this challenge by performing a simultaneous decomposition of different tensors. However, tensor factorization tends to suffer from a lack of robustness as the number of components affects the results to a large extent. In this work, a general framework for coupled tensor factorization is built to extract reliable components. Results from both individual and coupled decompositions are compared and divergence measures are used to adapt the number of components. It results in a joint decomposition method with (i) a variable number of components, (ii) shared and unshared components among tensors and (iii) robust components. Results on simulated data show a better modelling of the sources composing the datasets and an improved evaluation of the number of shared sources.**

I. Introduction

The rush for data collection over the past decade has led to an evolution of the way many datasets are represented. A flat matrix, limited to two dimensions, is no longer suitable in some applications where more complex abstractions are required. Social networks, biomedical, audio or imaging datasets, among others, now often have to deal with more than two dimensions. To better exploit the potential of these multi-dimensional datasets, higher-order tensors, i.e., multi-way arrays, have naturally taken over matrices. Taking more than two dimensions into account makes the analysis of the tensors both more informative and more challenging than in the case of matrices.

Tensor factorization (i.e., tensor decomposition) has emerged as one of the key tools to investigate these high-dimensional datasets. Generalizing matrix factorization techniques, it decomposes the complex tensors into more basic and interpretable parts [1]. From a blind source separation point of view, it can be seen as decomposing a mixture of sources (i.e., the tensor) into its constituent parts, named components. These components can be used for various purposes, from visualization to feature extraction or classification task. The actual objective is traditionally to find meaningful components with physical interpretation that are assumed to represent real sources composing the dataset. Applications that have successfully used tensor factorization techniques as a blind source separation method include audio and speech processing [2], chemometrics [3] or neuroscience [4].

As a natural extension of individual tensor decomposition, the problem of decomposing two or more tensors jointly is of great interest in many applications where more than one source of information is available. A joint analysis of different datasets that record similar phenomena has the potential to draw a more complete picture of the underlying structure of the data. This coupled tensor factorizations task requires one or more common dimensions between the two tensors, and utilizes these similar dimensions to couple the decompositions with an alternating optimization approach or, more recently, with an all-at-once joint optimization method [5]. Such a coupled analysis has already proved to be very insightful in various fields, from neuroscience [6] to collaborative filtering [7].

Despite its attractiveness, individual tensor factorization suffers from robustness issues that should be addressed to turn it into a reliable blind source separation method. Because every tensor decomposition method is mainly based on the modelling of as much variance of the dataset as possible, components can mix different sources composing the dataset to achieve this goal. This phenomenon is more frequent when many sources share similarities in one or more dimensions, such as in neuroscience (i.e., different sources may have common brain area, occurrence time or frequency band). While this situation could theoretically be avoided by specifying the right number of components, the development of efficient methods to estimate the ideal number of components remains an open challenge. Hence, components extracted are often inconsistent and affected to a large extent by the number of components chosen for the decomposition, altering their interpretation.

Another problem arising when performing coupled tensor analysis is the presence of both shared and unshared (i.e., dataset-specific) sources. This heterogeneity among the tensors amplifies the problem of mixed sources described above, as sources can also be mixed across the tensors.

These drawbacks limit the effectiveness of tensor factorization techniques and call for more robust methods. This work introduces an adaptive coupled tensor decompositions framework to produce more robust components. In the rest of this work, all the explanations are given for, but not limited to, two tensors that can be of different orders with at least one dimension in common.

978-0-9928-6265-7/16/$31.00 ©2016 IEEE

II. Methods

In its simplest and most common form (i.e. the PARAFAC decomposition [8]), tensor factorization approximates a tensor by a sum of components, where each component is modelled as the product of rank-one tensor, as represented in figure 1. For a tensor of order n, each component is formed by the outer product of n vectors, or factors, that can be seen as the signatures of this component in the n dimensions. These signatures are easy to visualize and can provide valuable insights into the data. The factors of all the components in one dimension form the factor matrix for this dimension.

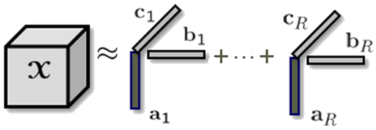


Fig. 1. Representation of a PARAFAC decomposition for a 3^{rd}-order tensor $\mathcal{X}$ with R components. Figure adapted from [9].

Coupled tensor decompositions addresses the problem of decomposing simultaneously different tensors that have common and non-common dimensions, as represented in figure 2. The underlying is that a source present in different tensors has identical signatures for the common dimensions in these tensors. Signatures in non-common dimensions complete the picture of the source. Hence, factor matrices in dimensions common to many tensors are derived using these many tensors while factor matrices in dimensions specific to one tensor are derived from this single tensor. Various challenges arise when performing such coupled analysis as the different tensors are often (i) of different orders, (ii) incomplete (i.e., with missing data) and (iii) have both shared and unshared sources.

The purpose of this work is to develop a framework to use the results from both individual and joint decompositions to extract more consistent components, i.e., components that are more likely to represent a single source. After a brief overview of the decomposition method used in section II-A, the proposed robust robust coupled tensor factorizations framework (i.e., RCTF) is developed in section II-B.

A. Coupled Matrix and Tensor Factorization algorithm

An efficient gradient-based optimization approach to perform the coupled decomposition task with tensors of different orders was introduced by Acar et al. [5]. This method, named coupled matrix and tensor factorization (i.e., CMTF), tackles both the challenge of tensors of different orders and of incomplete data. As an illustration, consider $\mathcal{X}$ and $\mathbf{Y}$, a 3^{rd}-order tensor and a matrix, respectively, with the first dimension in common. The cost function is defined as:

$$f(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{V}) = \| \mathcal{X} - [\![\mathbf{A},\mathbf{B},\mathbf{C}]\!] \|^2 + \| \mathbf{Y} - \mathbf{A}\mathbf{V}^\top \|^2 \quad (1)$$

where $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{V}$ are factor matrices with R columns, i.e., R components. Here, $[\![\mathbf{A},\mathbf{B},\mathbf{C}]\!]_{ijk} = \sum_{\ell=1}^{R} A_{i\ell}B_{j\ell}C_{k\ell}$ is used to denote the PARAFAC model. In [5], Acar et al. derived the gradient for these factor matrices, leading to an efficient gradient-based optimization algorithm for tensor decomposition. We have performed two modifications to the classic CMTF algorithm to make it more adapted for the framework developed.

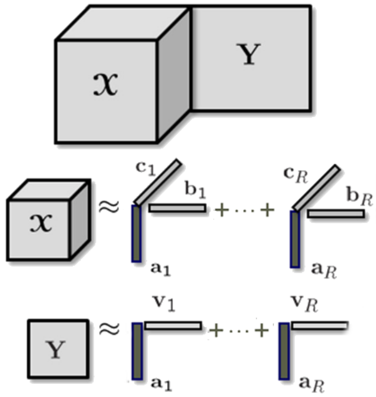


Fig. 2. Representation of a joint decomposition for a 3^{rd}-order tensor $\mathcal{X}$ and a matrix $\mathbf{Y}$ with R components. $\mathcal{X}$ and $\mathbf{Y}$ have one common dimension, leading to one factor matrix, composed of the a_i's, common for both $\mathcal{X}$ and $\mathbf{Y}$. Figure adapted from [9].

First, we have changed the algorithm to account for both shared and unshared sources. The first x components are decomposed in a joint way, using the classic CMTF algorithm with both tensors, while the remaining components are derived using only their corresponding tensor. This modification also enables to perform a coupled decomposition with a different number of components for each tensor, which we expect to be beneficial in many practical situations.

Second, we have added nonnegativity constraints on the factor matrices extracted. By their additive model which does not allow subtraction [10], nonnegative decompositions have proved to be effective to provide naturally sparse and more meaningful components. In order to make the CMTF algorithm nonnegative, the factor matrices in the cost function 1 are modelled as squared element-wise. The gradient can be derived in the same way as performed in [5]. Unlike the first modification, the nonnegativity constraint is not mandatory for the good behaviour of the algorithm.

B. RCTF Framework

The intuitive idea behind this work is to consider that a component resulting from an individual decomposition of a tensor $\mathcal{X}$ that properly models a source should stay the same in a joint decomposition with a tensor $\mathcal{Y}$. A component from tensor $\mathcal{X}$ that differs between the individual and the joint decompositions is assumed to be (i) affected by another source of $\mathcal{X}$ or by some noise in the individual decomposition, (ii) wrongly associated with a component of $\mathcal{Y}$ in the joint decomposition or (iii) associated to a component from $\mathcal{Y}$ that is itself affected by another source or some noise. The framework implemented is described in algorithm 1.

The easiest way to deal with case (ii) is to prevent the association between non-similar components to happen prior to a joint decomposition. This can be done by trading the traditional random initialization of a joint decomposition for a more sophisticated initialization method that draws information from separate decompositions and specify which components are shared and which are tensor-specific. Hence, separate decompositions of $\mathcal{X}$ and $\mathcal{Y}$ are performed (lines 4 to 10 in algorithm 1), with $N_{\mathcal{X}}$ and $N_{\mathcal{Y}}$ components, respectively. An $N_{\mathcal{X}} \times N_{\mathcal{Y}}$ similarity matrix between components is computed, based on a similarity measure and the common dimensions among tensors. Components presenting a similarity larger than a merging threshold th_m are merged (i.e., averaged in the common dimensions), forming the shared components in the upcoming joint decomposition. The remaining components of tensor $\mathcal{X}$ ($\mathcal{Y}$) that do not have any corresponding component in the decomposition of $\mathcal{Y}$ ($\mathcal{X}$) are kept tensor specific, i.e., derived only from $\mathcal{X}$ ($\mathcal{Y}$) in the joint decomposition, using the modification brought to CMTF described previously. As a result, only components which are assumed to mainly model the same source are derived jointly in the coupled tensor decomposition (line 13 in algorithm 1), addressing case (ii).

Cases (i) and (iii) can be detected after the joint decomposition by finding divergences between individual and joint decompositions (lines 14 to 24 in algorithm 1). In both cases, the ideal response should be the addition of a component to model this other source or noise, for tensor $\mathcal{X}$ in the case of (i) and for tensor $\mathcal{Y}$ in the case of (iii). To this purpose, results of the joint decomposition are compared to the results of previous individual decompositions, using a similarity measure and a convergence threshold th_c. If no significant divergence is found for the shared components, no issue is detected and the algorithm stops. Otherwise, an adjustment step is necessary to ensure a proper modelling (line 28 in algorithm 1).

The sources present in components that did not converge can be represented in at least two parts: the main source, denoted s, present in components of both tensors and responsible for the association of the components, and another source of variation n responsible for the non convergence. The underlying idea behind the adjustment step is to split the divergent components, denoted c, in two parts, one shared to correctly model s and another tensor-specific to model n. To this purpose, another joint decomposition is performed with a different initialization. Instead of merging all the identical components, components that did not converge after the first joint decomposition are initialized as two shared components a and b, coming from the individual decompositions of tensor $\mathcal{X}$ and $\mathcal{Y}$, respectively. The dimensions specific to tensor $\mathcal{X}$ ($\mathcal{Y}$) are initialized with the results of the previous joint decomposition for b (a). After the new joint decomposition, similarity between a and c is computed and compared to similarity between b and c. If component a (b) presents a higher similarity, it is assumed to model s while b (a) is assumed to model n. Hence, the source responsible for the divergence comes from tensor $\mathcal{Y}$ and the number of components to decompose $\mathcal{Y}$ is incremented by one. Results from the joint decomposition are then used to initialize separate decompositions and the whole process is repeated until convergence.

For more details about the implementation, the Matlab code is available at http://sites.uclouvain.be/absil/2016.07.

It results in both a more reliable and a more versatile joint decomposition method, with (i) a variable number of components, (ii) both shared and unshared components and (iii) robust components that are retrieved in both the individual and in the joint decompositions.

Algorithm 1 RCTF: Robust Coupled Tensor Factorization framework

```
Inputs: $\mathcal{X}$, $\mathcal{Y}$, $R_{\mathcal{X}}$, $R_{\mathcal{Y}}$, $th_m \in [0,1]$, $th_c \in [0,1]$
Output: $K_{\mathcal{X}}$, $K_{\mathcal{Y}}$
 1: ite ← 0
 2: while convergence = 0 do
 3:     ite ← ite +1
    Individual decompositions :
 4:     if ite = 1 then
 5:         $F_{\mathcal{X}}$ ← Decompose($\mathcal{X}$, $R_{\mathcal{X}}$)
 6:         $F_{\mathcal{Y}}$ ← Decompose($\mathcal{Y}$, $R_{\mathcal{Y}}$)
 7:     else
 8:         $F_{\mathcal{X}}$ ← Decompose($\mathcal{X}$, $R_{\mathcal{X}}$, initialize as $K_{\mathcal{X}}$)
 9:         $F_{\mathcal{Y}}$ ← Decompose($\mathcal{Y}$, $R_{\mathcal{Y}}$, initialize as $K_{\mathcal{Y}}$)
10:     end if
    Coupled decomposition :
11:     $D$ ← Closeness($F_{\mathcal{X}}$, $F_{\mathcal{Y}}$)
12:     $L$ ← Link_Components($D$, $th_m$)
13:     ($K_{\mathcal{X}}$, $K_{\mathcal{Y}}$) ← Decompose(($\mathcal{X}$, $\mathcal{Y}$), ($R_{\mathcal{X}}$, $R_{\mathcal{Y}}$), initialize
    as ($F_{\mathcal{X}}$, $F_{\mathcal{Y}}$) with $L$)
    Convergence analysis :
14:     n_shared_converg ← 0
15:     shared_diverg ← []
16:     for k in shared_components do
17:         $d_{\mathcal{X}\_k}$ ← Distance($F_{\mathcal{X}\_k}$, $K_{\mathcal{X}\_k}$)
18:         $d_{\mathcal{Y}\_k}$ ← Distance($F_{\mathcal{Y}\_k}$, $K_{\mathcal{Y}\_k}$)
19:         if $d_{\mathcal{X}\_k} > th_c$ and $d_{\mathcal{Y}\_k} > th_c$ then
20:             n_shared_converg ← n_shared_converg +1
21:         else
22:             shared_diverg ← [shared_diverg; k]
23:         end if
24:     end for
    End or Update :
25:     if n_shared_converg = n_shared_comp then
26:         convergence = 1
27:     else
28:         ($K_{\mathcal{X}}$, $K_{\mathcal{Y}}$, $R_{\mathcal{X}}$, $R_{\mathcal{Y}}$) ← Adjust($K_{\mathcal{X}}$, $K_{\mathcal{Y}}$, $R_{\mathcal{X}}$, $R_{\mathcal{Y}}$,
    shared_diverg)
29:     end if
30: end while
```

While the resulting algorithm can seem computationally expensive, the initializations of the decompositions with the results previously obtained largely decrease this complexity. In addition, the number of components is evolving along the iterations. Hence, the time complexity of this framework should not be compared with other standard methods without taking into account the time required to perform many simulations with different number of components for the other methods.

III. Results and Discussion

Tests are performed on simulated data, and results compared to state of the art coupled tensor decomposition methods, i.e., CMTF [5] and its generalization ACMTF [9]. These experiments are proof-of-concept tests that have the advantage of providing a ground truth against which to compare methods. Experiments on real EEG-MEG datasets are planned as further work.

A. Data Generation

The dataset generated consists in a 3^{rd}-order tensor and a 4^{th}-order tensor that have two dimensions in common. The size of all the dimensions is set to 40. Each tensor contains 12 sources, 9 of those being shared across tensors and 3 specific for each tensor. Sources are formed by randomly generating signatures in every dimensions and are normed to 1. Signatures of some components randomly picked are constrained to be identical in two dimensions, one common to both tensors and one specific to the 4^{th}-order tensor. Hence, some sources present similarities in one or two dimensions, modelling the problem of mixed sources described previously in section I. Some Gaussian noise is finally added to the dataset while ensuring the nonnegativity of the data.

B. Coupled tensor decomposition

Simulations are performed for 6, 9 and 12 initial components. The performance of our algorithm is evaluated through the modelling of the sources by the components and computed as $cos(angle) = norm(A.B)/(norm(A).norm(B))$, abbreviated as *angle* in the figure, with A the actual source and B the component modelling this source. Additionally, the number of unshared and shared components retrieved is displayed. For CMTF and ACMTF, a component extracted is considered unshared when its norm is at least ten times larger for one tensor than for the other. For our algorithm, merging and convergence thresholds th_m and th_c are set to 0.5 and 0.9. No optimization was performed to determine optimal values for these parameters. The similarity measure used is the $cos(angle)$ defined previously. The ACMTF algorithm uses a parameter σ to make the modelling of unshared components by the algorithm more likely [9]. Tests for ACMTF were repeated with 4 different values for σ: 10^{-4}, 10^{-3}, 10^{-2} and 10^{-1} and only results for the best value (i.e., 10^{-3}) are presented.

For every simulation, 50 runs were performed. To simplify the presentation, results from one run are averaged across the components retrieved. Hence, the 'modelling' for one run corresponds to the mean modelling of the components extracted in this run. All simulations are performed in Matlab.

Results are presented in figures 3 to 5, for CMTF, ACMTF and our method (RCTF). The histograms show the number of shared and specific components retrieved with each method, the density for x components representing the number of runs with x shared/specific components extracted divided by the total number of runs. Results show a higher number of components retrieved with our framework when the number of components is underestimated and a better evaluation of the number of shared components when the number of components is well approximated.

Boxplots show a better modelling of both the shared and unshared sources with our framework regardless the number of initial components used. The common dimensions of different sources lead to components mixing these sources in both CMTF and ACMTF. Our framework seems to be much more robust to this problem, while still dependent on the number of initial components chosen. The modelling of the unshared sources for CMTF with 6 initial components is not shown as the number of unshared sources retrieved is too small.

IV. Conclusion and Further Work

In this work, we have introduced a framework to perform coupled tensor decompositions based on the method developed by Acar et al. [5], and have shown its performance on numerical benchmarks with respect to standard coupled tensor decompositions methods. These improved performances originate from both a flexible way to deal with shared and unshared components and adjustments made across iterations to extract robust components. This work is a first step towards more reliable coupled tensor decomposition methods. Further work will include its validation on real EEG-MEG datasets the development of a more reliable and faster way to make the adjustment step in our framework.

Acknowledgment

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office. The first author is supported by a FRIA fellowship from F.R.S.-FNRS.

References

[1] A. Cichocki, D. Mandic, C. Caiafa, A.-H. Phan, G. Zhou, Q. Zhao, and L. De Lathauwer, *Tensor Decompositions for Signal Processing Applications*, IEEE Signal Process. Mag. 32: 145-163, 2015.

[2] Q. Wu, L.-Q. Zhang, G.-C. Shi, *Robust feature extraction for speaker recognition based on constrained nonnegative tensor factorization*, Journal of Computer Science and Technology, 25(4): 745-754, 2010.

[3] A. Smilde, R. Bro, and P. Geladi, *Multi-Way Analysis: Applications in the Chemical Sciences*, Wiley Publishing, 2004.

[4] C. Beckmann, and S. Smith, *Tensorial extensions of independent component analysis for multisubject fMRI analysis*, NeuroImage 25(1): 294-311, 2005.

[5] E. Acar, T. Kolda and D. Dunlavy, *All-at-once Optimization for Coupled Matrix and Tensor Factorizations*, KDD Workshop on Mining and Learning with Graphs, 2011.

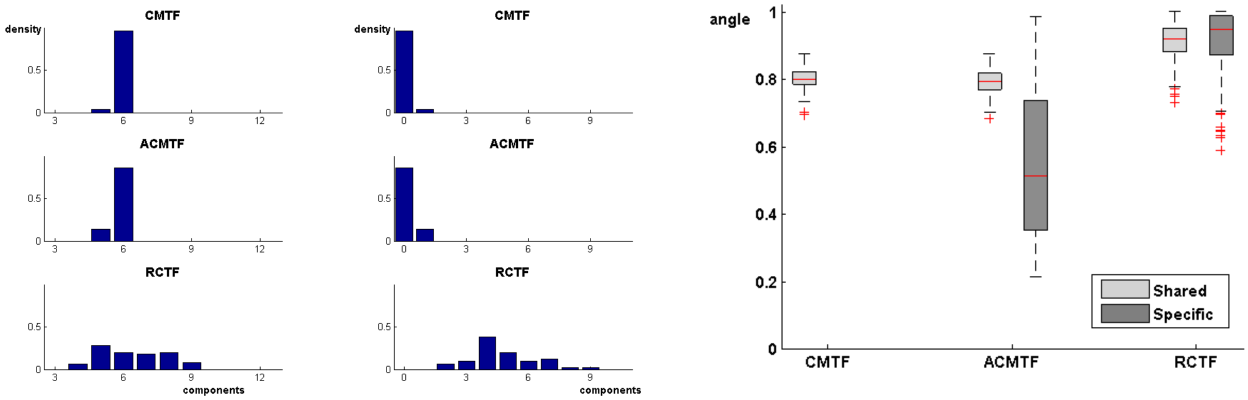
[6] E. Karahan, P. Rojas-Lopez, M. Bringas-Vega, P. Valdes-Hernandez, and P. Valdes-Sosa, *Tensor analysis and fusion of multimodal brain images*, Proc IEEE 103(9): 1531-1559, 2015.

[7] B. Ermis, E. Acar, and A.T. Cemgil, *Link prediction via generalized coupled tensor factorisation*, ECML/PKDD Workshop on Collective Learning and Inference on Structured Data, 2012.

[8] R.A. Harshman, *Foundations of the PARAFAC procedure: Models and conditions for an explanatory multimodal factor analysis*, UCLA Working Papers in Phonetics 16: 1-84, 1970.

[9] E. Acar, E. Papalexakis, G. Gurdeniz, M. Rasmussen, A. Lawaetz, M. Nilsson and R. Bro, *Structure-Revealing Data Fusion*, BMC Bioinformatics 15(1): 239, 2014.

[10] A. Cichocki, R. Zdunek, A.H. Phan, and S.-i. Amari, *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*, Wiley Publishing, section 1.2.1, 2009.

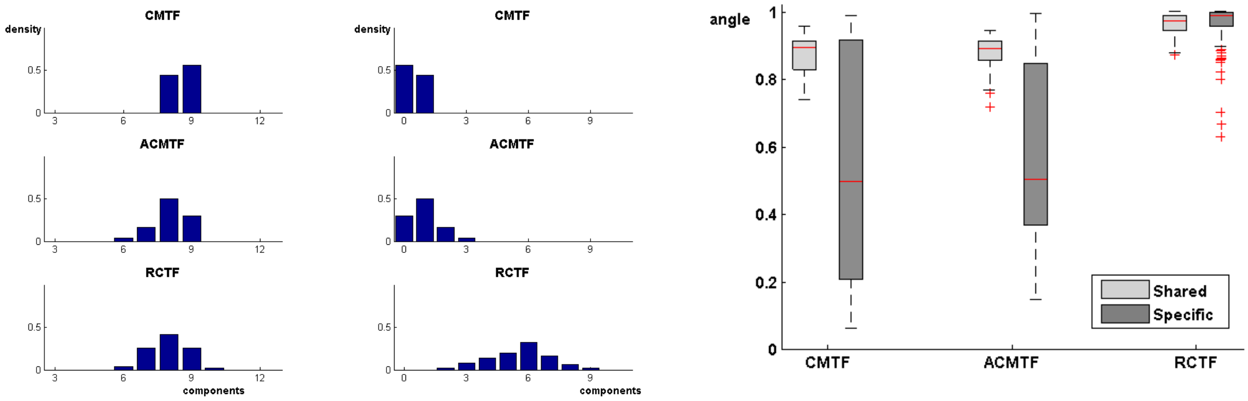


(a) Number of shared components retrieved

(b) Number of specific components retrieved

(c) Modelling of the shared and unshared sources

Fig. 3. Results of the simulations with 6 initial components for both tensors.

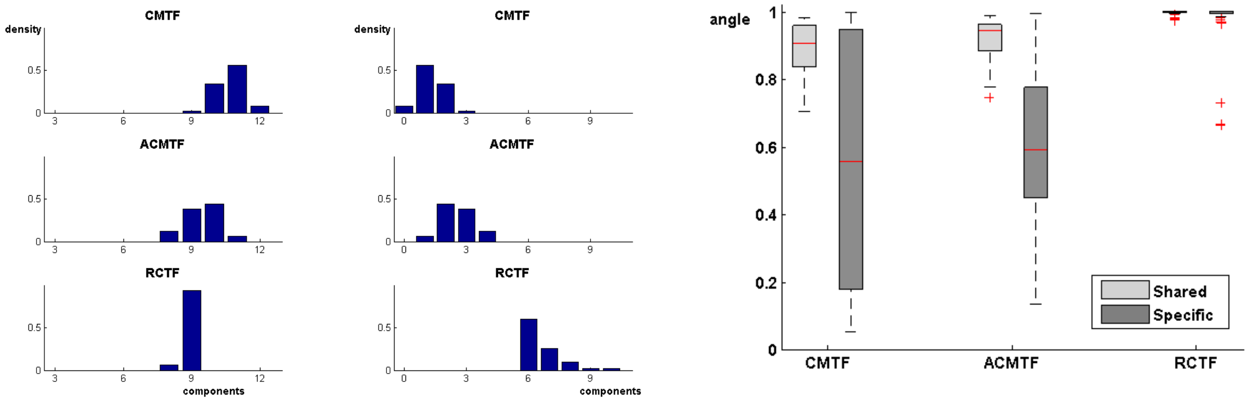


(a) Number of shared components retrieved

(b) Number of specific components retrieved

(c) Modelling of the shared and unshared sources

Fig. 4. Results of the simulations with 9 initial components for both tensors.



(a) Number of shared components retrieved

(b) Number of specific components retrieved

(c) Modelling of the shared and unshared sources

Fig. 5. Results of the simulations with 12 initial components for both tensors.