Direct ICA on Data Tensor via Random Matrix Modeling

Anonymous Author(s)
Affiliation
Address
email

Abstract

Independent component analysis (ICA) is a popular method for blind source separation (BSS). Classical ICA takes *data matrix* input formed by vector data. This paper focuses on ICA for BSS with third-order *data tensor* input formed by matrix data. Two approaches exist for this problem. The first reshapes matrix data into vectors to apply classical ICA, losing structural information. The second approach performs classical ICA twice by unfolding a data tensor into a data matrix along the row or column mode, partially preserving structures but with strong/ill BSS assumptions. This paper proposes a third approach via *random matrix* modeling, named as RAMICA. It works on data tensor *directly*, preserving row/column structures and having more general BSS assumptions. We progressively construct the RAMICA model and develop the RAMICA algorithm via defining new statistics for random matrix and new procedures for whitening and IC estimation. Experiments on both synthetic and real data show superior BSS performance of RAMICA over competing methods and provide insights on tradeoffs between various factors.

1 Introduction

Blind source separation (BSS) assumes that *observed data* are generated from unknown *latent sources*, and aims to recover these sources with the observations only. Independent component analysis (ICA) is a popular method for BSS [1]. Classical ICA treats P sources as random variables, and assumes they are mutually independent and linearly mixed to produce M observations $\{\underline{x_m}\}$ as:

$$\underline{x_m} = a_{m1}\underline{s_1} + \cdots + a_{mP}\underline{s_P}, \tag{1}$$

where the underscore indicates a random variable, $\underline{x_m}$ is the mth observation, $\underline{s_1}, \cdots, \underline{s_P}$ are the latent sources named as *independent components* (ICs), and $a_{m1}, \cdots, a_{mP}$ are mixing coefficients. Stacking random variables into vectors, we have the *vector-matrix notation* of (1) as:

$$\underline{\mathbf{x}} = \mathbf{A}\underline{\mathbf{s}} \in \mathbb{R}^M, \tag{2}$$

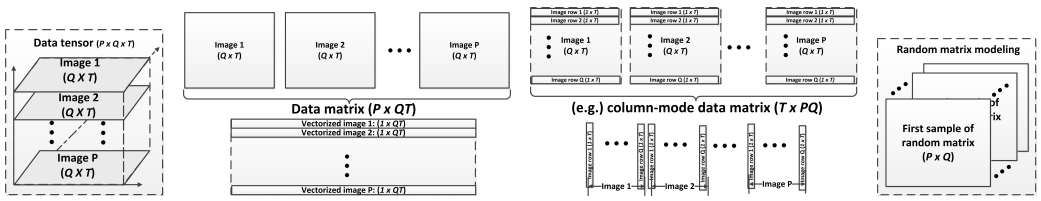
where $\underline{\mathbf{x}} = [\underline{x_1}, \cdots, \underline{x_M}]^\top$ and $\underline{\mathbf{s}} = [\underline{s_1}, \cdots, \underline{s_P}]^\top$ are observation (mixture) and source random vectors, and $\mathbf{A} = \{a_{mp}\} \in \mathbb{R}^{M\times P}$ is the unknown constant matrix, namely *mixing matrix*. It is usually assumed that $M = P$ so that $\mathbf{A} \in \mathbb{R}^{P\times P}$ is square, which we follow hereafter. Considering T available samples[1] of observations $\underline{\mathbf{x}}$ (sources $\underline{\mathbf{s}}$), align all samples in column to form the *data matrix* $\mathbf{X}$ (*source matrix* $\mathbf{S}$). We can then write the ICA model in $\mathbf{X}$ and $\mathbf{S}$ as:

$$\mathbf{X} = \mathbf{AS} \in \mathbb{R}^{P\times T}. \tag{3}$$

In other words, $\mathbf{X}$ contains T samples $\mathbf{x}(1)$, ..., $\mathbf{x}(T)$ of *random vector* $\underline{\mathbf{x}}$ in column, and $\mathbf{S}$ contains T samples of random vector $\underline{\mathbf{s}}$ in column. ICA aims to estimate the source matrix $\mathbf{S}$ and the mixing matrix $\mathbf{A}$ simultaneously with data matrix $\mathbf{X}$ as the only input [2].

[1] In this paper, the term 'sample' refers to samples of random variables, and the number of observations (P) is the number of data samples/examples in typical machine learning terminology.

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.



(a) Data tensor (b) Linear ICA (c) Mode-wise ICA (d) Proposed ICA

Figure 1: Given a $P \times Q \times T$ data tensor in (a), classical linear ICA vectorizes each image mixture into a long row vector and stacks them into a $P \times QT$ data matrix as in (b) for source recovery. Mode-wise ICA models, such as DTICA and MMICA, unfold the data tensor along a certain image mode, e.g., image column or row mode, to obtain a $T \times PQ$ (or $Q \times PT$) data matrix as in (c), on which to apply classical ICA (twice). The proposed RAMICA model deals with the data tensor directly without vectorization or unfolding as in (d).

Real-world data are often matrices or even higher-order tensors, such as multichannel EEG signals, images, videos, or social networks [3]. In such cases, all observed data of a particular problem form a *data tensor* $\boldsymbol{\mathcal{X}}$ with their natural multidimensional structures. Specifically, all observed matrix data form a third-order data tensor, while all observed Nth-order tensor form an $(N+1)$th-order data tensor. We can view such data tensor as *stacking* all data along a particular dimension (the first 'mode' by default). This paper focuses on *ICA for such data tensor input*. For convenience of discussion, we consider only *2D images* of size $Q \times T$ and we stack P of them into a third-order *data tensor* of size $P \times Q \times T$. Figure 1(a) shows the stacking of images into a data tensor.

There are two existing ICA approaches for data tensor input. The first *classical approach* is a linear one, which *vectorizes* (*reshapes*) images into vectors so that we can apply classical ICA methods such as FastICA [4], JADE [5], or Infomax [6]. Equivalently, we can view this *vectorization* process as *unfolding* the data tensor of size $P \times Q \times T$ into a data matrix of size $P \times QT$ along the first mode, as shown in Fig. 1(b) (with an enlarged version in supplementary material). The sources can be recovered as vectors first and then folded (reshaped) back to images (matrices). However, the vectorization breaks the original structure and leads to high-dimensional vectors, imposing significant theoretical and computational challenges. There are some other ICA variations [7, 8, 9] where more complicated data inputs are considered (e.g., images with known forming factors), but images are still represented as vectors and they degenerate to classical ICA under basic (simplified) settings [10].

The second approach is to do *mode-wise (linear) ICA* to partially preserve structural information and explore computational benefits. This approach is illustrated in Fig. 1(c), where effectively the data tensor is unfolded along each of the original image dimensions (image row or column) into data matrices of size $Q \times PT$ and $T \times PQ$ to apply classical ICA (twice). This brings an additional issue of mixing modeling, with two ways summarized below.

The first way of modeling in mode-wise ICA is a *multilinear-mixing model* as in directional tensor ICA (DTICA) [11, 12], where an image mixture is generated by one source matrix and two mode-wise mixing matrices (one for each mode). DTICA forms row and column directional images by shifting the rows/columns and then estimates two mixing matrices by mode-wise FastICA. Similarly, Virta *et al.* [13, 14] generalize JADE and FOBI [15] to mode-wise versions for data tensor input. Such multilinear-mixing models have an inherent limitation. They *cannot do BSS* to recover multiple matrix sources since they model a single source matrix only, which is hard to interpret in a BSS context.

The second way of modeling in mode-wise ICA is a *multilinear-source model* as in multilinear mode-wise ICA (MMICA) [10], where an image mixture is generated by two mode-wise source matrices via a multilinear mixing matrix. This model resembles the mixing model (1) more closely. However, it assumes the sources are rank one and constructed by mode-wise source matrices. Thus, although MMICA can do BSS, it can only recover rank-one sources due to its strong assumptions.

This paper proposes a new, third approach for ICA with data tensor input. Different from all existing works, we aim to recover general (not only rank-one) sources as the first approach can do while preserving multidimensional structure as the second approach. We do so by working with the original data tensor *directly* as shown in Fig. 1(d), *without vectorization or unfolding*. We develop our new method by considering *random matrix* in modeling so we name it RAndom Matrix ICA (RAMICA). In the RAMICA model, a random matrix consists of multiple random vectors. It assumes observed image data are generated by mixing source images with row-wise or column-wise structures. We make three major contributions in developing this RAMICA model and deriving the RAMICA algorithm:

1. With random matrix modeling, we propose RAMICA, a new ICA approach for data tensor input that deals with data tensor directly without vectorization or unfolding. Thus, RAMICA preserves multidimensional structure and can recover general source matrices in BSS.
2. We define new statistics of random matrix including covariance matrix, white matrix, independence, and higher-order cumulants, as the basis for developing RAMICA for data tensor input.
3. We formulate the RAMICA objective function by introducing a respective *whitening* step to obtain respective *whitened* random matrix and a new *cumulant operator* for random matrix. Then we derive a new RAMICA algorithm to recover source matrices with the Jacobi method.

We provide all proofs (of three lemmas and two theorems) in the supplementary material.

2 ICA via Random Matrix Modeling

Notations. Constants are in normal fonts, and random variables are underlined. Scalars, vectors, matrices, and tensors are denoted by lowercase, lowercase boldface, uppercase boldface, and bold calligraphic letters, respectively. E.g., their constant versions are x, $\mathbf{x}$, $\mathbf{X}$, and $\boldsymbol{\mathcal{X}}$, and their random versions are $\underline{x}$, $\underline{\mathbf{x}}$, $\underline{\mathbf{X}}$, and $\underline{\boldsymbol{\mathcal{X}}}$. The supplementary material has a list of symbols for easy reference.

Random vector. A random vector is a vector of random variables $\underline{\mathbf{x}} = [\underline{x_1}, \cdots, \underline{x_Q}]^\top$. Its expectation is a vector $E(\underline{\mathbf{x}}) = [E(\underline{x_1}), \cdots, E(\underline{x_Q})]^\top$. Its covariance matrix is

$$\Sigma(\underline{\mathbf{x}}) = \begin{bmatrix} \sigma^2(\underline{x_1}) & \cdots & cov(\underline{x_1}, \underline{x_Q}) \\ \vdots & & \vdots \\ cov(\underline{x_Q}, \underline{x_1}) & \cdots & \sigma^2(\underline{x_Q}, \underline{x_Q}) \end{bmatrix}, \tag{4}$$

where $cov(\underline{x_i}, \underline{x_j})$ is the covariance of $\underline{x_i}$ and $\underline{x_j}$, and $\sigma^2(\underline{x_i})$ is the variance of $\underline{x_i}$. If $\underline{x_1}, \cdots, \underline{x_Q}$ are mutually independent, $\underline{\mathbf{x}}$ is independent, and $\Sigma(\underline{\mathbf{x}})$ becomes diagonal. $E(\underline{\mathbf{x}})$ and $\Sigma(\underline{\mathbf{x}})$ are the first and second order *cumulants* of $\underline{\mathbf{x}}$. Higher order cumulants with order $r \geq 3$ are conventionally denoted by $[\mathcal{Q}_r(\underline{\mathbf{x}})]_{i_1 \cdots i_r}$ or $cum(\underline{x_{i_1}}, \cdots, \underline{x_{i_r}})$, where $i_1, \cdots, i_r$ are the mode-wise indices. See supplementary material for properties of cumulants.

Random matrix. A random matrix is a matrix of random variables $\underline{\mathbf{X}} = [\underline{x_{ij}}] \in \mathbb{R}^{P \times Q}$. Its expectation matrix is $E(\underline{\mathbf{X}}) = [E(\underline{x_{ij}})] \in \mathbb{R}^{P \times Q}$. Traditionally, its covariance matrix is defined as

$$\Sigma(\underline{\mathbf{X}}) := \Sigma(vec(\underline{\mathbf{X}})) \in \mathbb{R}^{(PQ)\times(PQ)}, \tag{5}$$

where $vec(\cdot)$ is the vectorization operator [16, 17, 18], and its higher order cumulants are defined via flattening the random matrix to the random vector. Thus, the independence of random matrix involves the independence of all elements. Equivalently, these definitions treat random matrix as its vectorized version, without considering any structural information.

RAndom Matrix ICA (RAMICA) model. Here, we follow (1), (2), and (3) in classical ICA.

Instead of (1), we have M mixtures $\underline{\mathbf{x}_1}, \cdots, \underline{\mathbf{x}_M} \in \mathbb{R}^Q$ of P random vector sources (ICs) as:

$$\underline{\mathbf{x}_m} = a_{m1}\underline{\mathbf{s}_1} + \cdots + a_{mP}\underline{\mathbf{s}_P}, \tag{6}$$

where $\underline{\mathbf{x}_m}$ is the mth mixture random vector, $\underline{\mathbf{s}_1}, \cdots, \underline{\mathbf{s}_P}$ are the independent source random vectors, and $a_{m1}, \cdots, a_{mP}$ are mixing coefficients. Again, we assume $M = P$. Stacking P random vectors $\{\underline{\mathbf{x}_m}\}$ and $\{\underline{\mathbf{s}_p}\}$ into random matrices along the first mode respectively, we obtain the *observation (mixture) matrix* $\underline{\mathbf{X}} = \left[\underline{\mathbf{x}_1}, \cdots, \underline{\mathbf{x}_P}\right]^\top \in \mathbb{R}^{P \times Q}$, and the source matrix $\underline{\mathbf{S}} = \left[\underline{\mathbf{s}_1}, \cdots, \underline{\mathbf{s}_P}\right]^\top \in \mathbb{R}^{P \times Q}$. We have the full matrix notation version of (6) instead of (2) below:

$$\underline{\mathbf{X}} = \mathbf{A}\underline{\mathbf{S}}, \tag{7}$$

where $\mathbf{A} \in \mathbb{R}^{P \times P}$ is the *mixing matrix* and assumed to be full-rank. With T samples of such *random matrices*, we form the data tensor $\boldsymbol{\mathcal{X}}$ and source tensor $\boldsymbol{\mathcal{S}}$ of size $P \times Q \times T$. Then we can write RAMICA model in data tensors $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{S}}$ instead of (3) in a *partial* Tucker decomposition [3] as

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{S}} \times_1 \mathbf{A}, \tag{8}$$

where $\times_1$ denotes the mode-1 multiplication [3] (see supplementary material for definition). The RAMICA objective is to estimate the source tensor $\boldsymbol{\mathcal{S}}$ and mixing matrix $\mathbf{A}$ given the data tensor $\boldsymbol{\mathcal{X}}$ only. Figure 1(d) shows this RAMICA model for data tensor, which is viewed along the third mode (instead of the first mode of stacking) as T samples of the random matrix $\underline{\mathbf{X}}$, i.e. $\mathbf{X}(1), \cdots, \mathbf{X}(T)$.

Remark: Note that (6), (7), and (8) in RAMICA correspond to (1), (2), and (3) in classical ICA, respectively. When $Q = 1$, the RAMICA model degenerates to the classical ICA model. Thus, RAMICA is a natural second-order generalization of classical ICA, without vectorization or unfolding. A key difference between (6) and (1) is that while $\underline{\mathbf{s}}_1, \cdots, \underline{\mathbf{s}}_P$ are independent, *the components of each source random vector $\underline{\boldsymbol{s}}_p$ can be dependent and encode structural information*. This allows structural information to be better preserved in RAMICA than in classical ICA.

RAMICA assumptions. We make three assumptions analogous to classical ICA [19]:

1. The expectation of mixture or source matrix is zero, i.e. $E(\underline{\mathbf{S}}) = E(\underline{\mathbf{X}}) = \mathbf{0}_{P\times Q}$. When this is not the case, we can do a *centering* preprocessing to subtract the mean.
2. The covariance matrix of the source random matrix is an identity matrix. Note the definition in (5) is equivalent to vectorizing $\underline{\mathbf{X}}$, leading to the first, vectorization-based ICA approach. Thus, we need *new definitions of related statistics for random matrix* to embody structural information.
3. At most one independent random vector from $\{\underline{\mathbf{s}}_p\}$ has multivariate Gaussian distribution.

New statistics of random matrix. We define new statistics for RAMICA via *tensor contraction*.

Definition 1. *The **covariance matrix** of a zero-mean random matrix $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_1, \cdots, \underline{\boldsymbol{x}}_P\right]^\top \in \mathbb{R}^{P\times Q}$ is defined as:*

$$\widetilde{\Sigma}(\underline{\boldsymbol{X}}) = \frac{1}{Q} E[\underline{\boldsymbol{X}}\underline{\boldsymbol{X}}^\top] \in \mathbb{R}^{P\times P}. \tag{9}$$

Each element of $\widetilde{\Sigma}(\underline{\boldsymbol{X}})$ is the covariance of two corresponding random vectors:

$$[\widetilde{\Sigma}(\underline{\boldsymbol{X}})]_{ij} = \widetilde{cov}(\underline{\boldsymbol{x}}_i, \underline{\boldsymbol{x}}_j) = \frac{1}{Q}\sum_{q=1}^{Q} cov(\underline{x}_{iq}, \underline{x}_{jq}), \tag{10}$$

where $cov(\cdot,\cdot)$ on the most right is the conventional covariance.

Definition 2. *Given a zero-mean random matrix $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_1, \cdots, \underline{\boldsymbol{x}}_P\right]^\top \in \mathbb{R}^{P\times Q}$, its **cumulant** of order $r \geq 3$ denoted by $[\widetilde{\mathcal{Q}_r}(\underline{\boldsymbol{X}})]_{i_1\cdots i_r}$ or $\widetilde{cum}(\underline{\boldsymbol{x}}_{i_1}, \cdots, \underline{\boldsymbol{x}}_{i_r})$ is defined as:*

$$[\widetilde{\mathcal{Q}_r}(\underline{\boldsymbol{X}})]_{i_1\cdots i_r} = \frac{1}{Q}\sum_{q=1}^{Q} cum(\underline{x}_{i_1 q}, \cdots, \underline{x}_{i_r q}), \tag{11}$$

where $i_1, \cdots, i_r \in \{1, \cdots, P\}$, and $cum(\cdot)$ is the conventional cumulant of a random vector. In particular, we denote a special case $\mathcal{K}_r(\underline{\boldsymbol{x}}_p) = \widetilde{cum}(\underline{\boldsymbol{x}}_p, \cdots, \underline{\boldsymbol{x}}_p)$ (r times).

Lemma 1. *The newly defined cumulants for random matrix $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_1, \cdots, \underline{\boldsymbol{x}}_P\right]^\top \in \mathbb{R}^{P\times Q}$ satisfies the properties of conventional cumulants for random vectors: (1) **symmetry**, (2) **linearity**, (3) **independence**, and (4) **vanishing Gaussian**.*

Definition 3. *A random matrix $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_1, \cdots, \underline{\boldsymbol{x}}_P\right]^\top \in \mathbb{R}^{P\times Q}$ is **independent** if the conditional distribution of $\underline{\boldsymbol{x}}_i$ given $\underline{\boldsymbol{x}}_j = \boldsymbol{x}$ does not depend on $\underline{\boldsymbol{x}}_j$ (i.e. $\underline{\boldsymbol{x}}_i$ and $\underline{\boldsymbol{x}}_j$ are mutually independent):*

$$f_{\underline{\boldsymbol{x}}_i|\underline{\boldsymbol{x}}_j}(\underline{\boldsymbol{x}}_i|\underline{\boldsymbol{x}}_j) = f_{\underline{\boldsymbol{x}}_i}(\underline{\boldsymbol{x}}_i). \tag{12}$$

Using these new statistics defined for random matrix, we can derive our RAMICA algorithm. For simpler notations, we keep using Σ, $\mathcal{Q}_r$, and cum instead of $\widetilde{\Sigma}$, $\widetilde{\mathcal{Q}}_r$, and $\widetilde{cum}$ hereafter.

Alternative forms of the RAMICA model. Given P images of size $A \times B$ (A rows and B columns), we have two ways to form a data tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{P\times Q\times T}$ in Eq. (8): (1) ***row-wise RAMICA*** (rRAMICA): each image row is treated as a random vector $\underline{\mathbf{x}}_m \in \mathbb{R}^B$ and all the A rows are considered A samples of $\underline{\mathbf{x}}_m$, i.e. $\mathbf{x}_m(1), \cdots, \mathbf{x}_m(A)$; thus, $Q = B$ and $T = A$. (2) ***column-wise RAMICA*** (cRAMICA): each image column is treated as a random vector $\underline{\mathbf{x}}_m \in \mathbb{R}^A$ and all the B columns are considered B samples of $\underline{\mathbf{x}}_m$, i.e. $\mathbf{x}_m(1), \cdots, \mathbf{x}_m(B)$; thus, $Q = A$ and $T = B$. rRAMICA and cRAMICA explore row and column structural information of image data, respectively. Real-world image data often have structures in both rows or columns. Therefore, both rRAMICA and cRAMICA can be effective and reveal different aspects of data. In other words, while there seems to be a 'mode selection' issue, from the other perspective, this offers *alternative explanations* of data that could be helpful in *interpretation*. For convenience of discussion, when we talk about RAMICA, we refer to **cRAMICA** unless specified explicitly.

Differences with existing methods. Here we highlight the key differences. The covariance matrice and cumulant of RAMICA have sizes of $P \times P$ and P^r, respectively, while their linear (vectorization-based) counterparts have much larger sizes of $PQ \times PQ$ and $(PQ)^r$ (typically $r = 4$ in ICA). Thus, they have much smaller computational and memory footprints than its linear counterparts. In terms of source separation capability, RAMICA captures structural information better than classical ICA, and lifts the restriction of rank-one sources in MMICA, while being superior over DTICA's single source assumption that is hard to interpret and unable to do source separation.

Extension to higher orders. Our model can be extended to higher-order data tensors via *random tensor modeling* and tensor contraction over multiple modes. For example, we now consider a fourth order tensor $\widetilde{\boldsymbol{\mathcal{X}}}$ of size $P \times Q \times R \times T$ formed by stacking P tensors of size $Q \times R \times T$. Our mixing model in (6) can be extended to $M = P$ mixtures $\underline{\mathbf{X}}_1, \cdots, \underline{\mathbf{X}}_M \in \mathbb{R}^{Q \times R}$ of P random matrix sources (ICs) as: $\underline{\mathbf{X}}_m = a_{m1}\underline{\mathbf{S}}_1 + \cdots + a_{mP}\underline{\mathbf{S}}_P$. We then view $\widetilde{\boldsymbol{\mathcal{X}}}$ as T samples of random tensors with size $P \times Q \times R$. We can subsequently define the *mode-wise* covariance matrice and cumulant for mode 1, now with summation over Q and R instead of just Q. Given P tensors of size $A \times B \times C$, we will have three ways to form a fourth-order data tensor, treating the mode-n slice as a random matrix. Further extensions (e.g., to fifth-order tensor of $P \times Q \times R \times S \times T$) can be similarly formulated.

3 The RAMICA algorithm

Given zero-mean input, the RAMICA algorithm has two steps, i.e. whitening and IC estimation.

Definition 4. *A random matrix $\underline{\boldsymbol{X}} = \left[\underline{\boldsymbol{x}}_1, \cdots, \underline{\boldsymbol{x}}_P\right]^\top \in \mathbb{R}^{P \times Q}$ is **white** if its covariance matrix is an identity matrix:*

$$\Sigma(\underline{\boldsymbol{X}}) = \boldsymbol{I}_{P \times P}. \tag{13}$$

Thus, the ***whitening*** step of the RAMICA model (7) is to find a matrix $\mathbf{W}$ such that $\Sigma(\mathbf{W}\underline{\mathbf{X}}) = \mathbf{I}_{P \times P}$.

Theorem 1. *For the RAMICA model* (7)*, let $\boldsymbol{W}$ denote any inverse square root of $\Sigma(\underline{\boldsymbol{X}})$, i.e. $[\Sigma(\underline{\boldsymbol{X}})]^{-1/2}$. Then $\boldsymbol{W}$ is the whitening matrix and $\boldsymbol{W}\underline{\boldsymbol{X}}$ is white.*

For simpler notation, we keep using $\underline{\mathbf{X}}$ as the *whitened* random matrix in the remaining of this paper.

Whitened RAMICA model. After RAMICA whitening, we have $\mathbf{W}\underline{\mathbf{X}} = \mathbf{WA}\underline{\mathbf{S}} = \mathbf{U}\underline{\mathbf{S}}$. Since $\Sigma(\mathbf{W}\underline{\mathbf{X}}) = \mathbf{I}_{P \times P} = \mathbf{U}\Sigma(\underline{\mathbf{S}})\mathbf{U}^\top = \mathbf{U}\mathbf{U}^\top$, $\mathbf{U}$ is orthogonal. The *whitened RAMICA* model can be rewritten as

$$\underline{\mathbf{X}} = \mathbf{U}\underline{\mathbf{S}}, \tag{14}$$

where $\mathbf{U}$ is the *whitened mixing matrix*.

Lemma 2. *Given the whitened RAMICA model* (14)*, the fourth-order cumulants of (whitened) $\underline{\boldsymbol{X}}$ satisfy:*

$$[\mathcal{Q}_4(\underline{\boldsymbol{X}})]_{ijkl} = \sum_p u_{ip}u_{jp}u_{kp}u_{lp}\mathcal{K}_4(\underline{\boldsymbol{s}}_p), \forall i, j, k, l \in \{1, \cdots, P\}. \tag{15}$$

RAMICA IC estimation. It is difficult to recover sources directly from cumulants. Instead, we can convert the BSS problem to a matrix diagonalization problem as in JADE [5], via a cumulant-based mapping of the whitened mixing matrix $\mathbf{U}$. To do this, we define a new cumulant operator first.

Definition 5. *Given the whitened RAMICA model* (14)*, the **cumulant operator** $\mathscr{F}_{\underline{\boldsymbol{X}}}$ is defined by the fourth-order cumulant tensor $\mathcal{Q}_4(\underline{\boldsymbol{X}})$ of random matrix $\underline{\boldsymbol{X}}$ as:*

$$\mathscr{F}_{\underline{\boldsymbol{X}}} : \boldsymbol{M} \in \mathbb{R}^{P \times P} \mapsto [\mathscr{F}_{\underline{\boldsymbol{X}}}(\boldsymbol{M})]_{ij} = \textstyle\sum_{k,l} m_{kl}[\mathcal{Q}_4(\underline{\boldsymbol{X}})]_{ijkl} \in \mathbb{R}^{P \times P}. \tag{16}$$

Lemma 3. *Given the whitened RAMICA model* (14)*, the cumulant operator $\mathscr{F}_{\underline{\boldsymbol{X}}}$ satisfies:*

$$[\mathscr{F}_{\underline{\boldsymbol{X}}}(\boldsymbol{M})]_{ij} = \sum_p u_{ip}u_{jp}\mathcal{K}_4(\underline{\boldsymbol{s}}_p)\sum_{k,l} m_{kl}u_{kp}u_{lp}, \forall i, j \in \{1, \cdots, P\}. \tag{17}$$

Theorem 2. *Given the whitened model* (14)*, the matrix $\boldsymbol{U}^\top\mathscr{F}(\boldsymbol{M})\boldsymbol{U}$ is diagonal for $\forall \boldsymbol{M} \in \mathbb{R}^{P \times P}$.*

RAMICA objective. Theorem 2 reveals the connection between the whitened RAMICA model (14) and $\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})$. We can take a set of matrices $\mathbf{M}_i$ and make the matrix set $\{\mathbf{U}^\top\mathcal{F}_{\underline{\mathbf{X}}}(\mathbf{M}_i)\mathbf{U}\}$ as diagonal as possible for BSS. In practice, they cannot be made exactly diagonal because the model does not hold exactly and there are sampling errors. In fact, the diagonality of a symmetric matrix $\mathbf{Q} = \mathbf{U}^\top\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})\mathbf{U}$ can be measured by the sum of the squares of off-diagonal entries: $\sum_{i \neq j} q_{ij}^2$ [20]. Since for a given matrix $\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})$, the square sum over all elements of the matrix is preserved under an orthogonal transformation, minimizing the sum of squares of off-diagonal elements is equivalent to maximizing the sum of squares of diagonal elements [21]. We formulate our objective function based on this

property. For a set of basis matrix $\{\mathbf{M}_i \in \mathbb{R}^{P\times P}\}$, we maximize the following objective function with respect to orthogonal matrix $\mathbf{U}$:

$$\sum_i ||diag(\mathbf{U}^\top \mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M}_i)\mathbf{U})||^2, \tag{18}$$

where $||diag(\cdot)||^2$ denotes the sum of squares of the diagonal elements. In this paper, we use the standard basis of $\mathbb{R}^{P\times P}$, i.e. $\{\mathbf{E}^{ij} = \mathbf{e}_i\mathbf{e}_j^\top\}_{i,j=1}^P$, which can reduce computational cost significantly.

RAMICA algorithm. Similar to JADE, we apply Jacobi method to optimize (18) to compute the whitened mixing matrix $\mathbf{U}$. Specifically, to use the Jacobi method, we first align the set of matrices $\{\mathscr{F}(\mathbf{E}^{ij})\}_{i,j=1}^P$ into an extended matrix $\mathbf{M} = [\mathscr{F}(\mathbf{E}^{11}), \cdots, \mathscr{F}(\mathbf{E}^{PP})]$. Then, we apply the Jacobi method on $\mathbf{M}$ to conduct a series of Jacobi rotations, each of which handles two rows and two columns at a time [22]. After this, we obtain $\mathbf{U}^{-1}$ by multiplying the rotation matrices. Subsequently, we get $\mathbf{A}$ from $\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{W}$. The RAMICA algorithm is summarized in Algorithm 1, where we view $\boldsymbol{\mathcal{X}}$ as T samples of $\underline{\mathbf{X}}$ to obtain sample-based estimates of various statistics. The theoretical properties of the newly defined cumulant can be extended to such empirical estimations following [19, §2.7]. Note computing higher-order statistics like cumulants are computationally expensive in general. Nonetheless, with a tensor-based formulation, most of the computations can be parallelized to improve the scalability, particularly with the availability of parallel and cloud computing.

Algorithm 1 RAMICA: ICA via Random Matrix Modeling

1: **Input:** a zero-mean data tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{P\times Q\times T}$.
2: **Whitening**: viewing $\boldsymbol{\mathcal{X}}$ as T samples of $\underline{\mathbf{X}}$, compute the sample estimate of the whitening matrix $\mathbf{W}$ according to Theorem 1, and compute the whitened data tensor: $\boldsymbol{\mathcal{X}} \times_1 \mathbf{W}$.
3: **IC Estimation:**
4: Compute the 4th-order cumulant tensor from Eq. (11).
5: Compute cumulant matrices $\{\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{E}^{ij})\}$ according to Eq. (16) for $\forall i, j \in \{1, \cdots P\}$.
6: Align the above matrices to form matrix $\mathbf{M} = [\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{E}^{11}), \cdots, \mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{E}^{PP})]$.
7: Apply Jacobi method on $\mathbf{M}$ to obtain rotation matrices.
8: Get $\mathbf{U}^{-1}$ by multiplying these Jacobi rotation matrices.
9: Compute $\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{W}$, and invert $\mathbf{A}^{-1}$ to get $\mathbf{A}$.
10: Compute $\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{X}} \times_1 \mathbf{A}^{-1}$.
11: **Output:** the mixing matrix $\mathbf{A}$ and source tensor $\boldsymbol{\mathcal{S}}$.

4 Experiments

In this section, we perform evaluations on BSS with data tensor input for sources having general 2D structures, rather than special structures such as rank one. We compare two versions of RAMICA, i.e. rRAMICA and cRAMICA, against classical ICA methods FastICA, JADE, and Infomax using the codes provided by the authors. The mode-wise ICA with a *multilinear-mixing model* (DTICA) cannot do BSS because it assumes only one source matrix, while the one with a *multilinear-source model* (MMICA) can only recover rank-one sources by design. Therefore, both DTICA and MMICA will fail on this more general setting of BSS with data tensor input.

We conduct experiments on both synthetic and real data. For synthetic data, we generate *column-wise* source random vectors for mixing. With only column structural information synthesized, we expect cRAMICA to perform well whereas rRAMICA not. For real data, we linearly mix natural images and then recover them from their mixtures. Natural images are expected to have both row and column structural information. Hence, both rRAMICA and cRAMICA are expected to recover the sources to some extent. Which one does better could depend on whether row or column structure is stronger.

For BSS performance measurement, we use the popular Amari error [23] calculated over the demixing matrices (i.e., the inverse of the mixing matrices). For convenience of presentation, we report Amari error values multiplied by 100 throughout this paper. We report the average performance with standard deviations (std) over *100* repetitions for each experimental setting below.

4.1 Blind source separation on synthetic data

We first study how well RAMICA can recover sources from synthetic data tensor generated according to the column-wise RAMICA model (8).

Data generation. To simulate *column-wise* structural information in each random vector $\underline{\mathbf{s}} \in \mathbb{R}^Q$, only its first component $\underline{s_1}$ is randomly generated, while its other components have the following linear relationships with $\underline{s_1}$:

$$\underline{s_q} = \alpha\underline{s_1} + \beta(q-1), \tag{19}$$

where $\alpha \sim \mathcal{N}(1,1)$ (Gaussian distribution) and $\beta \sim U(0,1)$ (uniform distribution) are randomly generated, and $q \in \{2, \cdots, Q\}$. Thus, $\{\underline{s_1}, \cdots, \underline{s_Q}\}$ are dependent rather than independent and $\underline{\mathbf{s}}$ is a column random vector with column structures. We consider the following four distributions that generate the first component of each source random vector $\underline{\mathbf{s}_p} \in \mathbb{R}^Q$ for $p \in \{1, \cdots, P\}$:

- Psn: Pearson distribution with zero mean, unit variance, unit skewness, and unit kurtosis.
- Stu: Student-t distribution with 5 degrees of freedom.
- Exp: Exponential distribution with $\lambda = 1$.
- Lap: Laplace distribution with $\mu = 0$ and $b = 1/\sqrt{2}$.

Following the above generation, T samples of the pth random vector $\mathbf{s}_p$ form the pth source image $\mathbf{S}_p = [\mathbf{s}_p(1), \cdots, \mathbf{s}_p(T)] \in \mathbb{R}^{Q \times T}$. Finally, stacking P source images along the first mode, we have the source tensor $\boldsymbol{\mathcal{S}} = [\mathbf{S}_1; \cdots; \mathbf{S}_P] \in \mathbb{R}^{P \times Q \times T}$. Such sources are much more realistic/general than the restricted rank-one sources synthesized in MMICA [10]. Note although rank-one sources can be combined to produce low-rank (or high-rank) sources, there is great indeterminacy.

In generating the mixing matrix $\mathbf{A}$, we need to guarantee its invertibility. We generate $\mathbf{A}$ in three steps: (i) uniformly generate a $P \times P$ matrix with each entry between zero and one; (ii) normalize the generated matrix by column; (iii) add an identity matrix to the one in (ii). With $\boldsymbol{\mathcal{S}}$ and $\mathbf{A}$ generated, we further generate $\boldsymbol{\mathcal{X}}$ according to (8).

Design factors. In simulations, we have the following design factors investigated with several choices:

- $\mathcal{D} = \{Psn, \boldsymbol{Stu}, Exp, Lap\}$: the distribution used to generate the (first components of) sources.
- $P \in \{2, \mathbf{4}, 8, 16\}$: the number of sources.
- $Q = \{16, \mathbf{32}, 64, 128\}$: the dimension of random vectors.
- $T = \{16, 32, \mathbf{64}, 128\}$: the number of samples for random matrices.
- $\sigma^2 = \{\mathbf{0}, 0.01, 0.02, \cdots, 0.1, 0.15, 0.2\}$: the noise level of Gaussian noise that is added to the observation as:
$$\underline{\mathbf{X}} = \mathbf{A}\underline{\mathbf{S}} + \underline{\mathbf{E}}, \tag{20}$$
where $\underline{\mathbf{E}}$ denotes standard Gaussian noise $vec(\underline{\mathbf{E}}) \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$. The default setting is noise-free.

When studying one factor, we vary it with other factors fixed to their default settings in bold above.

Effect of $\mathcal{D}$. Table 1 reports the performance with varying $\mathcal{D}$ and default settings of other factors. All methods obtain different results for different $\mathcal{D}$s. Their performance indicates how challenging each $\mathcal{D}$ is. Almost all methods get the best results on Exp but the worst results on Psn, indicating Psn is more challenging than Exp. Among the five ICA methods, cRAMICA consistently achieves the best performance, though this is expected due to the *column-wise* data tensor generation. In particular, cRAMICA improves over JADE (the second best) by 31.4% on average. On the other hand, it is not surprising that rRAMICA gives poorer results. Nonetheless, real-world data often have both row-wise and column-wise structures so both rRAMICA and cRAMICA can be effective and reveal different aspects of data. This will be confirmed in the real data experiments.

Table 1: Effect of $\mathcal{D}$ for synthetic BSS. Other factors use default settings. Amari errors are reported and each entry is the mean±std of 100 repetitions. The best (second-best) Amari errors are highlighted in bold (underline).

$\mathcal{D}$	Infomax	FastICA	JADE	cRAMICA	rRAMICA
Psn	10.03±1.71	7.07±1.47	<u>6.90±1.61</u>	**5.12±2.11**	7.63±1.23
Stu	9.35±1.48	6.48±1.41	<u>6.33±1.44</u>	**4.45±2.09**	6.88±1.39
Exp	9.04±1.50	6.03±1.22	<u>5.93±1.19</u>	**3.72±1.73**	6.73±1.24
Lap	9.29±1.44	6.37±1.42	<u>6.22±1.37</u>	**4.12±1.82**	6.60±1.19

Effect of P. Table 2 reports the performance with respect to P. Although all ICA methods deteriorate as P increases, cRAMICA consistently achieves the best performance and outperforms others by a large margin. E.g., cRAMICA outperforms JADE by 29.70% for $P = 4$. Again, rRAMICA is inferior to cRAMICA, but it outperforms Infomax.

Table 2: Effect of P for synthetic BSS shown as in Table 1.

P	Infomax	FastICA	JADE	cRAMICA	rRAMICA
2	1.97±0.65	1.51±0.53	1.47±0.54	**0.88±0.77**	<u>1.42±0.33</u>
4	9.35±1.48	6.48±1.41	<u>6.33±1.44</u>	**4.45±2.09**	6.88±1.39
8	36.58±4.29	25.36±5.07	<u>24.48±4.53</u>	**20.92±5.53**	26.71±3.92
16	134.43±10.07	103.24±12.95	<u>99.36±12.36</u>	**92.52±14.87**	106.00±11.21

Effect of Q. Table 3 reports the results on various Qs. cRAMICA still gives the best results. In addition, both rRAMICA and cRAMICA slightly deteriorate as Q increases. In contrast, classical ICA methods behave differently. Infomax behaves similarly as RAMICA but FastICA and JADE achieve slightly better performance with increasing Q. Such difference could be due to the trade-off between the benefits of having more samples QT and the detriments of more dependent samples. The detriments dominate for Infomax but the benefits dominate for JADE and FastICA.

Table 3: Effect of Q for synthetic BSS shown as in Table 1.

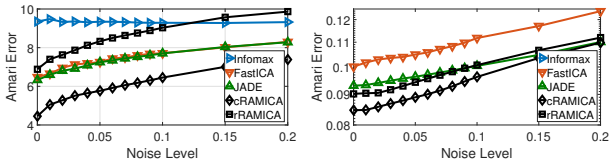
Q	Infomax	FastICA	JADE	cRAMICA	rRAMICA
16	8.80±1.63	6.56±1.72	<u>6.33±1.68</u>	**4.29±2.20**	6.76±1.43
32	9.35±1.48	6.48±1.41	<u>6.33±1.44</u>	**4.45±2.09**	6.88±1.39
64	9.79±1.50	6.37±1.17	<u>6.28±1.28</u>	**4.58±2.09**	6.92±1.29
128	10.09±1.44	6.26±1.12	<u>6.20±1.21</u>	**4.72±2.09**	6.93±1.19

Effect of T. Table 4 reports the results on Ts. Again, cRAMICA performs the best. All methods have better performance with increasing T, where the improvement of cRAMICA is the most significant, as shown in the last row. The improvements for classical ICA methods are less significant because they have to compensate the detriments of more dependent samples. Though the benefits of larger sample size dominate, the detriments of more dependent samples reduce their improvement rate. Comparing the results in Tables 3 and 4, we can also see that T has a larger effect on the performance of RAMICA than Q.

Table 4: Effect of T for synthetic BSS shown as in Table 1. The last row reports the improvement rate from $T = 16$ to $T = 128$.

T	Infomax	FastICA	JADE	cRAMICA	rRAMICA
16	10.81±1.69	7.59±1.66	7.42±1.58	**6.51±2.36**	8.01±1.24
32	9.91±1.39	7.55±1.39	7.13±1.25	**5.71±2.01**	7.59±1.33
64	9.35±1.48	6.48±1.41	6.33±1.44	**4.45±2.09**	6.88±1.39
128	9.08±1.58	5.87±1.13	5.78±1.14	**3.19±1.47**	6.47±0.98
↑	16.00%	22.66%	22.10%	51.00%	19.26%

Effect of σ^2. In the last study on synthetic BSS, we examine the sensitivity of these ICA methods with respect to noise as shown in Fig. 2(a). We can see that cRAMICA and rRAMICA have similar sensitivity to noise with JADE and FastICA. It may be because they are all based on the fourth-order cumulants. Infomax is the least sensitive to noise but it performs the worst in most cases. In addition, cRAMICA largely outperforms the others in this experiment.



(a) Synthetic BSS. (b) Real image BSS.

Figure 2: Effect of σ^2 for synthetic and real image BSS. Other factors for synthetic BSS use default settings. The image BSS experiment has four 256×256 source images. The average Amari errors over 100 repetitions are reported. The Infomax results in real image BSS are all above the chosen upper limit so they are not visible.

4.2 Blind image separation

We further perform evaluations on real-world image data. Natural image data tend to have structures in both row and column. Thus, both rRAMICA and cRAMICA should work to some extent. Which one performs better will depend on whether row or column structure dominates. Next, we conduct real data experiments to verify this.

Data. Source images are taken from the Caltech256 repository [24]. We selected 4,424 images with strong higher-order statistics from the total 30,607 images for blind image separation experiments. All selected images are resized to a standard size of 256×256 with 256 gray levels so $Q = T = 256$.

Experimental settings. We repeat the following process 100 times: 1) randomly select four source images ($P = 4$); 2) mix them using a mixing matrix randomly generated in the same way as in the synthetic study to produce four mixture images according to (8) (equivalent to the classical ICA model (3)); 3) recover the sources from the four mixtures by ICA methods; 4) compute the Amari errors accordingly. In addition, we add noise to the mixing process to do sensitivity study. The average Amari errors are reported.

Image separation results. Figure 2(b) shows the recovery performance across different noise levels. The results of Infomax are above the chosen upper limit so they are not shown in the figure for clarity. We can see both rRAMICA and cRAMICA can obtain better BSS performance than the other methods when the noise level is below 0.1. This confirms that real-world images contain both row and column structures and both rRAMICA and cRAMICA have their merits. Furthermore, the observation that cRAMICA outperforms rRAMICA indicates that the column structure is stronger than the row structure on average for the selected Caltech256 images. Nonetheless, when the noise level increases, the performance gain of cRAMICA and rRAMICA over JADE diminishes. Thus, a future direction could be to improve their robustness against noise.

5 Conclusion

This paper proposed a new ICA method RAMICA for BSS with data tensor input. It differs from the classical vectorization-based approach and more recent mode-wise approach by dealing with data tensor directly, without vectorization or unfolding. Thus, it can do more general BSS while preserving structural information. We build RAMICA based on random matrix modeling with two versions: row-wise and column-wise RAMICA. By defining new statistics of random matrix, we develop a two-step RAMICA algorithm with a new cumulant operator and the Jacobi method. Experimental results on both synthetic and real image BSS showed that RAMICA outperformed competing ICA methods greatly in BSS on data tensor, with its two versions having their respective merits.

References

[1] A. Hyvärinen. Independent component analysis: Recent advances. *Philosophical Transactions of the Royal Society of London: Mathematical, Physical and Engineering Sciences*, 371(1984), 2013.

[2] A. Hyvärinen and E. Oja. Independent component analysis: Algorithms and applications. *IEEE Transactions on Neural Networks*, 13(4-5):411–430, 2000.

[3] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos. *Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data*. CRC Press, 2013.

[4] A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9:1483–1492, 1997.

[5] J. F. Cardoso and A. Souloumiac. Blind beamforming for non-gaussian signals. *IEE Processings F (Radar and Signal Processing)*, 140(6):362–370, 1993.

[6] A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

[7] M. A. O. Vasilescu and D. T.erzopoulos. Multilinear independent components analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 547–553, 2005.

[8] M. A. O. Vasilescu and D. T.erzopoulos. Multilinear (tensor) ICA and dimensionality reduction. In *Conference on Independent Component Analysis and Signal Separation*, pages 818–826, 2007.

[9] R. G. Raj and A. C. Bovik. MICA: A multilinear ICA decomposition for natural scene modeling. *IEEE Transactions on Image Processing*, 17(3):259–271, 2008.

[10] H. Lu. Learning modewise independent components from tensor data using multilinear mixing model. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 288–303, 2013.

[11] L. Zhang, Q. Gao, and D. Zhang. Directional independent component analysis with tensor representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2008.

[12] Q. Gao, L. Zhang, D. Zhang, and H. Xu. Independent components extraction from image matrix. *Pattern Recognition Letters*, 31(3):171–178, 2010.

[13] J. Virta, B. Li, K. Nordhausen, and H. Oja. Jade for tensor-valued observations, 2016.

[14] J. Virta, B. Li, K. Nordhausen, and H. Oja. Independent component analysis for tensor-valued data, 2016.

[15] J. F. Cardoso. Source separation using higher order moments. In *Proc. ICASSP*, pages 2109–2112, 1989.

[16] M. Bilodeau and D. Brenner. *Theory of Multivariate Statistics*. Springer-Verlag New York, 1999.

[17] S. S. Muni, N. Tatjana, and R. Dietrich. Models with a Kronecker product covariance structure: estimation and testing. Technical report, Swedish University of Agricultural Sciences, 2007.

[18] M. John. *Multivariate Statistics: Old School*. CreateSpace Independent Publishing Platform, Department of Statistics, University of Illinois at Urbana-Champaign, 2015.

[19] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, 2001.

[20] P. Comon. Tensor diagonalization, a useful tool in signal processing. *IFAC Symposium on System Identification*, 1:77–82, 1994.

[21] G. Deco and D. Obradovic. *An Information-Theoretic Approach to Neural Computing (1st Edition)*. Springer Series in Perspectives in Neural Computing, 1996.

[22] B. D. Clarkson. A least squares version of algorithm as 211: The F-G diagonalization algorithm. *Applied Statistics*, 37:317–321, 1988.

[23] S. Amari, A. Cichocki, and H. H. Yang. A new learning algorithm for blind signal separation. In *Neural Information Processing Systems (NIPS)*, pages 757–763, 1996.

[24] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset, 2006. California Institute of Technology.

[25] P. Ilmonen, H. Oja, and R. Serfling. On invariant coordinate system (ICS) functionals. *International statistical review*, 80:93–110, 2012.

Supplementary Material

Direct ICA on Data Tensor via Random Matrix Modeling

NIPS2017 – Paper 659

List of symbols.

Table 5: List of symbols with descriptions and types.

Symbol	Description	Type
$\mathbf{A}$	the mixing matrix	constant matrix
$cum(\cdots)$	the conventional rth order cumulant tensor of random vector $\underline{\mathbf{x}}$	statistic operator
$\widetilde{cum}(\cdots)$	our new rth order cumulant tensor of random matrix $\underline{\mathbf{X}}$	statistic operator
$\mathscr{F}_{\underline{\mathbf{X}}}(\cdot)$	the new cumulant operator	statistic operator
$\mathcal{K}_r(\underline{\mathbf{x}}_p)$	$\widetilde{cum}(\underline{\mathbf{x}}_p, \cdots, \underline{\mathbf{x}}_p)$, a special case of $\widetilde{\mathcal{Q}}_r(\underline{\mathbf{X}})$	statistic operator
M	the number of observations	constant scalar
P	the number of sources	constant scalar
$\widetilde{\mathcal{Q}}_r(\underline{\mathbf{X}})$	$\widetilde{cum}(\underline{\mathbf{x}}_{i_1}, \cdots, \underline{\mathbf{x}}_{i_r})$	statistic operator
$\mathcal{Q}_r(\underline{\mathbf{x}})$	$cum(\underline{x_{i_1}}, \cdots, \underline{x_{i_r}})$	statistic operator
r	the order of higher order cumulants	constant scalar
$\boldsymbol{\mathcal{S}}$	the source data tensor	constant tensor
$\mathbf{S}$	the source data matrix	constant matrix
$\underline{\mathbf{S}}$	the source random matrix	random matrix
$\mathbf{S}(t)$	the tth sample of $\underline{\mathbf{S}}$	constant matrix
$\underline{\mathbf{s}}$	the source random vector	random vector
$\underline{\mathbf{s}}_p$	the pth source vector	random vector
$\underline{s_p}$	the pth source scalar	random scalar
$\underline{\mathbf{s}}(t)$	the tth sample of $\underline{\mathbf{s}}$	constant vector
$\widetilde{\Sigma}(\underline{\mathbf{X}})$	our new covariance matrix of random matrix $\underline{\mathbf{X}}$	statistic operator
$\Sigma(\underline{\mathbf{x}})$	the conventional covariance matrix of random vector $\underline{\mathbf{x}}$	statistic operator
T	the number of random samples	constant scalar
$\mathbf{U}$	the whitened mixing matrix	constant matrix
$\mathbf{W}$	the whitening matrix	constant matrix
$\boldsymbol{\mathcal{X}}$	the observation data tensor	constant tensor
$\mathbf{X}$	the observation data matrix	constant matrix
$\underline{\mathbf{X}}$	the original/whitened observation random matrix	random matrix
$\mathbf{X}(t)$	the tth sample of $\underline{\mathbf{X}}$	constant matrix
$\underline{\mathbf{x}}$	the observation random vector	random vector
$\underline{\mathbf{x}}_m$	the mth observation vector	random vector
$\underline{x_m}$	the mth observation scalar	random scalar
$\underline{\mathbf{x}}(t)$	the tth sample of $\underline{\mathbf{x}}$	constant vector

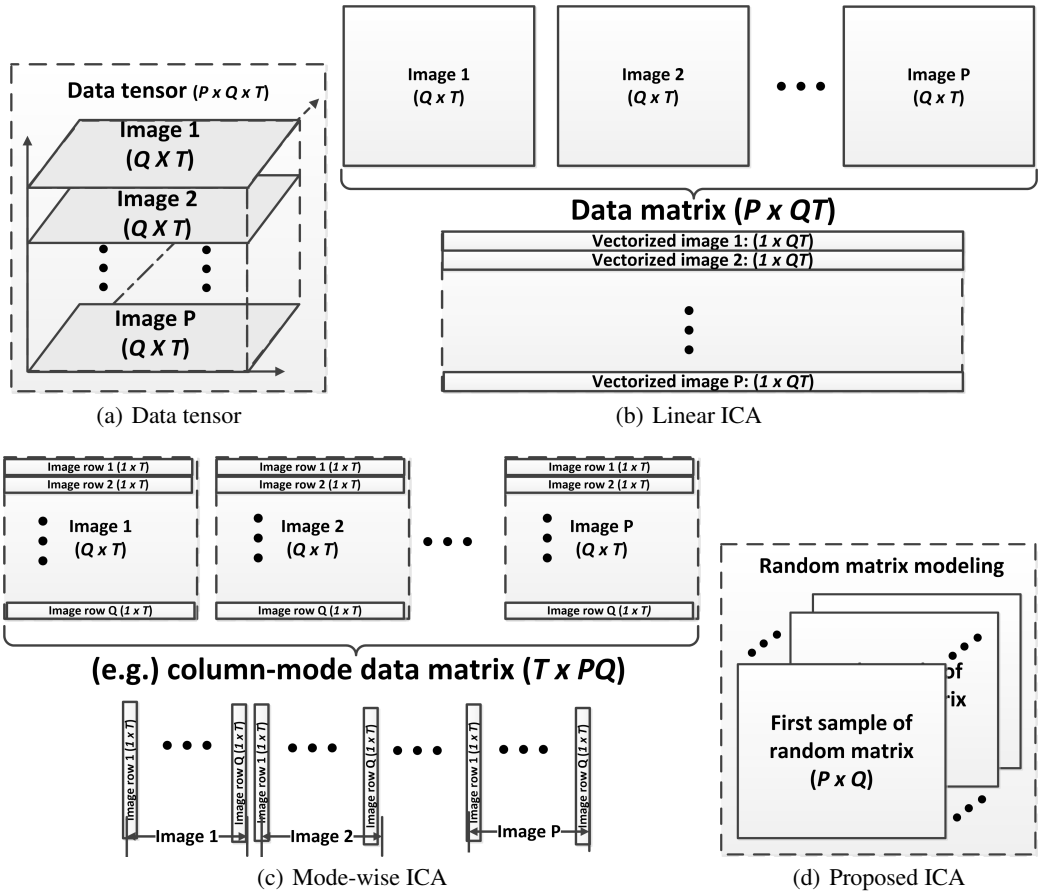
For simpler notations, Σ, $\mathcal{Q}_r$, and cum are used in place of $\widetilde{\Sigma}$, $\widetilde{\mathcal{Q}}_r$, and $\widetilde{cum}$ after definition.

Figure 1 in double resolution.

Figure 1 is reproduced in Fig. 3 with double resolution to show details better.

Properties of cumulants

Cumulants of random vector $\underline{\mathbf{x}}$ have the following properties: (1) ***symmetry***: $[\mathcal{Q}_r(\underline{\mathbf{x}})]_{i_1 \cdots i_r} = [\mathcal{Q}_r(\underline{\mathbf{x}})]_{i_{\sigma(1)} \cdots i_{\sigma(r)}}$ for any permutation $\sigma(\cdot)$; (2) ***linearity***: $cum(\underline{x_1}, \cdots, \underline{x_i} + \underline{y}, \cdots, \underline{x_r}) = cum(\underline{x_1}, \cdots, \underline{x_i}, \cdots, \underline{x_r}) + cum(\underline{x_1}, \cdots, \underline{y}, \cdots, \underline{x_r})$ and $cum(\underline{x_1}, \cdots, \alpha\,\underline{x_i}, \cdots, \underline{x_r}) = \alpha\, cum(\underline{x_1}, \cdots, \underline{x_i}, \cdots, \underline{x_r})$ for any random variable $\underline{y}$ and constant α; (3) ***independence***: if $\exists p, q \in \{1, \cdots, r\}$ where $\underline{x_{i_p}}$ and $\underline{x_{i_q}}$ are independent, then $[\mathcal{Q}_r(\underline{\mathbf{x}})]_{i_1 \cdots i_r} = 0$; (4) ***vanishing Gaussian***: if $\underline{\mathbf{x}}$ is Gaussian, $[\mathcal{Q}_r(\underline{\mathbf{x}})]_{i_1 \cdots i_r} = 0$ for any order $r \geq 3$.



(a) Data tensor (b) Linear ICA

(c) Mode-wise ICA (d) Proposed ICA

Figure 3: Given a $P \times Q \times T$ data tensor in (a), classical linear ICA vectorizes each image mixture into a long row vector and stacks them into a $P \times QT$ data matrix as in (b) for source recovery. Mode-wise ICA models, such as DTICA and MMICA, unfold the data tensor along a certain image mode, e.g., image column or row mode, to obtain a $T \times PQ$ (or $Q \times PT$) data matrix as in (c), on which to apply classical ICA (twice). The proposed RAMICA model in (d) deals with the data tensor directly without vectorization or unfolding.

Tensor *mode*-1 *product*

The *mode*-1 *product* of a third-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ by a matrix $\mathbf{U} \in \mathbb{R}^{J_n \times I_n}$, denoted by $\mathcal{A} \times_1 \mathbf{U}$, is a tensor with entries:

$$(\mathcal{A} \times_1 \mathbf{U})_{j_1 i_2 i_3} = \sum_{i_1} \mathcal{A}_{i_1 i_2 i_3} \cdot \mathbf{U}_{j_1 i_1}. \tag{21}$$

ICA procedures

ICA has three standard steps. (1) *Centering*: remove the first-order statistics from the data by shifting the sample mean to the origin. (2) *Whitening*: remove the second-order statistics from the data to obtain *whitened* variables. (3) *ICA Estimation*: use higher-order statistics of the data to estimate ICs. It is the core step of ICA, and different methods do it differently.

Proof of Lemma 1

This lemma can be obtained by using the corresponding properties of cumulants for random vector multiple times and do a final average operation as denoted in Eq. (11).

Proof of Theorem 1

Perform singular value decomposition (SVD) on the mixing matrix $\mathbf{A} = \mathbf{EDV}$, where $\mathbf{E}$ and $\mathbf{V}$ are orthogonal, and $\mathbf{D}$ is diagonal. Compute the covariance matrix of $\underline{\mathbf{X}}$ using the above decomposition:

$$\Sigma(\underline{\mathbf{X}}) = \frac{1}{Q} E[\mathbf{EDVSS}^\top \mathbf{V}^\top \mathbf{DE}^\top] = \mathbf{ED}^2\mathbf{E}^\top. \tag{22}$$

The second equality is due to *Assumption 2* of RAMICA and orthogonality of $\mathbf{V}$. Denote by $\mathbf{M} \in \mathbb{R}^{P\times P}$ any *orthogonal* matrix, then $\mathbf{W}$ can be written as:

$$\mathbf{W} = \mathbf{M}\mathbf{D}^{-1}\mathbf{E}^\top \in \mathbb{R}^{P\times P}, \tag{23}$$

according to [25].

Next, we calculate the covariance matrix of $\mathbf{W}\underline{\mathbf{X}}$ as

$$\begin{aligned} \Sigma(\mathbf{W}\underline{\mathbf{X}}) &= \tfrac{1}{Q} E[\mathbf{M}\mathbf{D}^{-1}\mathbf{E}^T\underline{\mathbf{X}}\underline{\mathbf{X}}^\top\mathbf{E}\mathbf{D}^{-1}\mathbf{M}^\top] \\ &= \mathbf{M}\mathbf{D}^{-1}\mathbf{E}^\top[\Sigma(\underline{\mathbf{X}})]\mathbf{E}\mathbf{D}^{-1}\mathbf{M}^\top \\ &= \mathbf{M}\mathbf{D}^{-1}\mathbf{E}^\top[\mathbf{E}\mathbf{D}^2\mathbf{E}^\top]\mathbf{E}\mathbf{D}^{-1}\mathbf{M}^\top = \mathbf{I}_{P\times P}. \end{aligned} \tag{24}$$

Therefore, we have proved that $\mathbf{W}\underline{\mathbf{X}}$ is white and $\mathbf{W}$ is the whitening matrix.

A commonly used whitening matrix

Similar to classical ICA, we can conduct eigenvalue decomposition on the covariance matrix as

$$\Sigma(\underline{\mathbf{X}}) = \mathbf{E}\mathbf{\Lambda}\mathbf{E}^\top, \tag{25}$$

where $\mathbf{E} = [\mathbf{e}_1, \cdots, \mathbf{e}_P]$ has the unit-norm eigenvectors as columns, and the diagonal matrix $\mathbf{\Lambda}$ consists of the eigenvalues. According to Theorem 1, the whitening matrix is

$$\mathbf{W} = \mathbf{\Lambda}^{-1/2}\mathbf{E}^\top. \tag{26}$$

Proof of Lemma 2

Calculate the fourth-order cumulants according to Lemma 1. We have:

$$\begin{aligned} [\mathcal{Q}_4(\underline{\mathbf{X}})]_{ijkl} &= cum(\textstyle\sum_p u_{ip}\underline{\mathbf{s}_p}, \sum_{p'} u_{jp'}\underline{\mathbf{s}_{p'}}, \sum_q u_{kq}\underline{\mathbf{s}_q}, \sum_{q'} u_{lq'}\underline{\mathbf{s}_{q'}}) \\ &= \textstyle\sum_{p,p',q,q'} u_{ip}u_{jp'}u_{kq}u_{lq'}cum(\underline{\mathbf{s}_p}, \underline{\mathbf{s}_{p'}}, \underline{\mathbf{s}_q}, \underline{\mathbf{s}_{q'}}). \end{aligned} \tag{27}$$

Due to the independence of $\{\underline{\mathbf{s}_p}\}$, only those products with $p = p' = q = q'$ are nonzero. Therefore, we have proved that

$$[\mathcal{Q}_4(\underline{\mathbf{X}})]_{ijkl} = \sum_p u_{ip}u_{jp}u_{kp}u_{lp}\mathcal{K}_4(\underline{\mathbf{s}_p}). \tag{28}$$

Proof of Lemma 3

This lemma can be proved by substituting cumulants $[\mathcal{Q}_4(\underline{\mathbf{X}})]_{ijkl}$ in Eq. (16) by the derived Eq. (15) of Lemma 2.

Proof of Theorem 2

According to matrix multiplication rules and Lemma 3, we know for $\forall i, j \in \{1, \cdots, P\}$:

$$\begin{aligned} [\mathbf{U}^\top\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})\mathbf{U}]_{ij} &= \textstyle\sum_{p,q} u_{pi}u_{qj}[\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})]_{pq} \\ &= \textstyle\sum_{p,q} u_{pi}u_{qj}\sum_m u_{pm}u_{qm}\mathcal{K}_4(\underline{\mathbf{s}_m})\sum_{k,l} m_{kl}u_{km}u_{lm} \\ &= \textstyle\sum_m \mathcal{K}_4(\underline{\mathbf{s}_m})\sum_{k,l} m_{kl}u_{km}u_{lm}\sum_p u_{pi}u_{pm}\sum_q u_{qj}u_{qm}. \end{aligned} \tag{29}$$

Since $\mathbf{U}$ is orthogonal, we have

$$[\mathbf{U}^\top\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})\mathbf{U}]_{ij} = \sum_m \mathcal{K}_4(\underline{\mathbf{s}_m})\sum_{k,l} m_{kl}u_{km}u_{lm}\delta_{im}\delta_{jm}. \tag{30}$$

Only those products with $i = j = m$ are nonzero. Thus, we have

$$[\mathbf{U}^\top\mathscr{F}_{\underline{\mathbf{X}}}(\mathbf{M})\mathbf{U}]_{ij} = \begin{cases} 0, & \text{if } i \neq j \\ \mathcal{K}_4(\underline{\mathbf{s}_i})\sum_{k,l} m_{kl}u_{ki}u_{li}, & \text{if } i = j \end{cases} \tag{31}$$

Therefore, we have proved the diagonality of $\mathbf{U}^\top\mathscr{F}(\mathbf{M})\mathbf{U}$.