Overcomplete Independent Component Analysis via SDP

Anastasia Podosinnikova
MIT

Amelia Perry
MIT

Alexander S. Wein
Courant Institute, NYU

Francis Bach
INRIA, ENS

Alexandre d'Aspremont
CNRS, ENS

David Sontag
MIT

Abstract

We present a novel algorithm for overcomplete independent components analysis (ICA), where the number of latent sources k exceeds the dimension p of observed variables. Previous algorithms either suffer from high computational complexity or make strong assumptions about the form of the mixing matrix. Our algorithm does not make any sparsity assumption yet enjoys favorable computational and theoretical properties. Our algorithm consists of two main steps: (a) estimation of the Hessians of the cumulant generating function (as opposed to the fourth and higher order cumulants used by most algorithms) and (b) a novel semidefinite programming (SDP) relaxation for recovering a mixing component. We show that this relaxation can be efficiently solved with a projected accelerated gradient descent method, which makes the whole algorithm computationally practical. Moreover, we conjecture that the proposed program recovers a mixing component at the rate $k < p^2/4$ and prove that a mixing component can be recovered with high probability when $k < (2-\varepsilon)p \log p$ when the original components are sampled uniformly at random on the hypersphere. Experiments are provided on synthetic data and the CIFAR-10 dataset of real images.

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

1 Introduction

Independent component analysis (ICA) models a p-dimensional *observation* x as a linear combination of k latent mutually independent *sources*:

$$x = D\alpha, \tag{1}$$

where $\alpha := (\alpha_1, \dots, \alpha_k)^\top$ and $D \in \mathbb{R}^{p \times k}$. The linear transformation D is called the *mixing matrix* and is closely related to the *dictionary matrix* from dictionary learning (see, e.g., Chen and Donoho, 1994; Chen et al., 1998). Given a sample $X := \{x^{(1)}, \dots, x^{(n)}\}$ of n observations, one is often interested in estimating the latent mixing matrix D and respective latent representations, $\alpha^{(1)}, \dots, \alpha^{(n)}$, also known as *sources*, of every observation.

A classical motivating example for ICA is the cocktail party problem, where one is interested in separating individual speakers' voices from noisy recordings. Here, each record is an observation and each speaker is an independent source. In general, ICA is a simple single-layered neural network and is widely used as an unsupervised learning method in machine learning and signal processing communities (see, e.g., Hyvärinen et al., 2001; Comon and Jutten, 2010).

There are three conceptually different settings of the ICA problem: (a) *complete*, or *determined*, where the dimension of observations coincides with the number of sources, i.e., $p = k$; (b) *undercomplete*, or *overdetermined*, with fewer sources than the dimension, i.e., $k < p$; and (c) *overcomplete*, or *underdetermined*, with more sources than the dimension, i.e., $k > p$. While the first two cases are well studied, the last one is more difficult and we address it here.

In the *complete* setting, where $k = p$, ICA is usually solved via pre-whitening of the data so that the whitened observations, $z := Wx$, are uncorrelated and all have unit variance, i.e., $\mathrm{cov}(z) = W\mathrm{cov}(x)W^\top = I$,

where W denotes the whitening matrix. Substituting $x = D\alpha$, we get $(WD)(WD)^\top = I$ which implies that the matrix $Q := WD$ is *orthogonal* and therefore the problem of finding the mixing matrix D boils down to finding the "correct" orthogonal matrix Q. Numerous "correctness" criteria, such as maximizing non-Gaussianity of sources, were proposed and respective algorithms for complete ICA are well known (see, e.g., Hyvärinen et al., 2001; Comon and Jutten, 2010). The most widely known complete ICA algorithms are possibly the *FastICA* algorithm by Hyvärinen (1999) and the *JADE* algorithm by Cardoso and Souloumiac (1993). This naturally extends to the undercomplete setting where one looks for an orthonormal matrix, where columns are orthogonal, instead. However, although nothing prevents us from whitening data in the overcomplete setting, the orthogonalization trick cannot be extended to the *overcomplete* setting, where $k > p$, since the mixing matrix D has more columns than rows and therefore cannot have full column rank.

Improvements in feature learning are among the advantages of overcomplete representations: it has been shown by Coates et al. (2011) that dense and overcomplete features can significantly improve performance of classification algorithms. However, advantages of overcomplete representations go far beyond this task (see, e.g., Bengio et al., 2013).

Originally, the idea of overcomplete representations was developed in the context of dictionary learning, where an overcomplete dictionary, formed by Fourier, wavelet, Gabor or other filters, is given and one is only interested in estimating the latent representations α. Different approaches were proposed for this problem including the method of frames (Daubechies, 1988) and basis pursuit (Chen and Donoho, 1994; Chen et al., 1998). Later in sparse coding, the idea of estimating a dictionary matrix directly from data was introduced (Olshausen and Field, 1996, 1997) and was shortly followed by the first overcomplete ICA algorithm (Lewicki and Sejnowski, 2000).[1] Further overcomplete ICA research continued in several fairly different directions based on either (a) various sparsity assumptions (see, e.g., Teh et al., 2003) or on (b) prior assumptions about the sources as by Lewicki and Sejnowski (2000) or (c) instead in a more general dense overcomplete setting (see, e.g., Hyvärinen, 2005; Comon and Rajih, 2006; De Lathauwer et al., 2007; Goyal et al., 2014; Bhaskara et al., 2014a,b; Anandkumar et al., 2015; Ma et al., 2016). Since we focus on this more general dense setting, we do not review or compare to the literature in the other settings.

[1] Recall the close relation between ICA and sparse coding: indeed, the maximum likelihood estimation of ICA with the Laplace prior on the sources (latent representations α) is equivalent to the standard sparse coding formulation with the ℓ_1-penalty.

In particular, we focus on the following problem: *Estimate the mixing matrix D given an observed sample $X := \{x^{(1)}, \dots, x^{(n)}\}$ of n observations.* We aim at constructing an algorithm that would bridge the gap between algorithms with theoretical guarantees and ones with practical computational properties. Notably, our algorithm does not depend on any probabilistic assumptions on the sources, except for the standard independence and non-Gaussianity, and the uniqueness of the ICA representation (up to permutation and scaling) is the result of the independence of sources rather than sparsity. Here we only focus on the estimation of the latent mixing matrix and leave the learning of the latent representation for future research (note that one can use, e.g., the mentioned earlier dictionary learning approaches).

Different approaches have been proposed to address this problem. Some attempt to relax the hard orthogonality constraint in the whitening procedure with more heuristic quasi-orthogonalization approaches (see, e.g., Le et al., 2011; Arora et al., 2012). Other approaches try to specifically address the structure of the model in the overcomplete setting (see, e.g., Hyvärinen, 2005; Comon and Rajih, 2006; De Lathauwer et al., 2007; Goyal et al., 2014; Bhaskara et al., 2014a,b; Anandkumar et al., 2015; Ma et al., 2016) by considering higher-order cumulants or derivatives of the cumulant generating function. The algorithm that we propose is the closest to the latter type of approach.

We make two conceptual contributions: (a) we show how to use second-order statistics instead of the fourth and higher-order cumulants, which improves sample complexity, and (b) we introduce a novel semi-definite programming-based approach, with a convex relaxation that can be solved efficiently, for estimating the

Algorithm 1 OverICA

1: **Input: Observations** $X := \{x_1, \dots, x_n\}$ **and latent dimension** k.
Parameters: The regularization parameter μ and the number s of generalized covariances, $s > k$.
2: **STEP I. Estimation of the subspace** W**:**
Sample vectors $t_1, \dots, t_s$.
Estimate matrices $H_j := \mathcal{C}_x(t_j)$ for all $j \in [s]$.
3: **STEP II. Estimation of the atoms:**
Given $G^{(i)}$ for every deflation step $i = 1, 2, \dots, k$:
Solve the relaxation (12) with $G^{(i)}$.
(OR: Solve the program (9) with $G^{(i)}$.)
Estimate the i-th mixing component d_i from B^*.
4: **Output: Mixing matrix** $D = (d_1, d_2, \dots, d_k)$.

columns of D. Overall, this leads to a computationally efficient overcomplete ICA algorithm that also has theoretical guarantees. Conceptually, our work is similar to the fourth-order only blind identification (FOOBI) algorithm (De Lathauwer et al., 2007), which we found to work well in practice. However, FOOBI suffers from high computational and memory complexities, its theoretical guarantee requires all kurtoses of the sources to be positive, and it makes the strong assumption that certain fourth-order tensors are linearly independent. Our approach resolves these drawbacks. We describe our algorithm in Section 2 and experimental results in Section 3.

2 Overcomplete ICA via SDP

2.1 Algorithm overview

We focus on estimating the latent mixing matrix $D \in \mathbb{R}^{p\times k}$ of the ICA model (1) in the overcomplete setting where $k > p$. We first motivate our algorithm in the population (infinite sample) setting and later address the finite sample case.

In the following, the i-th column of the mixing matrix D is denoted as d_i and called the i-th **mixing component**. The rank-1 matrices $d_1 d_1^\top, \dots, d_k d_k^\top$ are referred to as **atoms**.[2]

Our algorithm, referred to as **OverICA**, consists of two major steps: (a) construction of the **subspace W** spanned by the atoms, i.e.,

$$W := \operatorname{Span}\left\{d_1 d_1^\top, \dots, d_k d_k^\top\right\}, \tag{2}$$

and (b) estimation of individual atoms $d_i d_i^\top$, $i \in [k]$, given any basis of this subspace.[3] We summarize this high level idea[4] in Algorithm 1. Note that although the definition of the subspace W in (2) is based on the latent atoms, in practice this subspace is estimated from the known observations x (see Section 2.3). However, we do use this explicit representation in our theoretical analysis.

In general, there are different ways to implement these two steps. For instance, some algorithms implement the first step based on the fourth or higher order cumulants (see, e.g., De Lathauwer et al., 2007; Goyal et al., 2014). In contrast, we estimate the subspace W from the Hessian of the cumulant generating function which has better computational and sample complexities (see Section 2.3). Our algorithm also works (without any adjustment) with other implementations of the first step, including the fourth-order cumulant based one, but other algorithms cannot take advantage of our efficient first step due to the differences in the second step.

In the second step, we propose a novel semi-definite program (SDP) for estimation of an individual atom given the subspace W (Section 2.4.1). We also provide a convex relaxation of this program which admits efficient implementation and introduces regularization to noise which is handy in practice when the subspace W can only be estimated approximately (Section 2.4.2). Finally, we provide a deflation procedure that allows us to estimate all the atoms (Section 2.4.3). Before proceeding, a few assumptions are in order.

2.2 Assumptions

Due to the inherent permutation and scaling unidentifiability of the ICA problem, it is a standard practice to assume, without loss of generality, that

Assumption 2.1. *Every mixing component has unit norm, i.e.,* $\|d_i\|_2 = 1$ *for all* $i \in [k]$.

This assumption immediately implies that all atoms have unit Frobenius norm, i.e., $\left\|d_i d_i^\top\right\|_F = \|d_i\|_2^2 = 1$ for all $i \in [k]$.

Since instead of recovering mixing components d_i as in (under-) complete setting we recover atoms $d_i d_i^\top$, the following assumption is necessary for the identifiability of our algorithm:

Assumption 2.2. *The matrices (atoms)* $d_1 d_1^\top$, $d_2 d_2^\top$, $\dots$, $d_k d_k^\top$ *are linearly independent.*

This in particular implies that the number of sources k cannot exceed $m := p(p+1)/2$, which is the latent dimension of the set of all symmetric matrices $\mathcal{S}_p$. We also assume, without loss of generality, that the observations are centred, i.e., $\mathbb{E}(x) = \mathbb{E}(\alpha) = 0$.

2.3 Step I: Subspace Estimation

In this section, we describe a construction of an orthonormal basis of the subspace W. For that, we first construct matrices $H_1, \dots, H_s \in \mathbb{R}^{p\times p}$, for some s, which span the subspace W. These matrices are obtained from the Hessian of the cumulant generating function as described below.

Generalized Covariance Matrices. Introduced for complete ICA by Yeredor (2000), a generalized covariance matrix is the Hessian of the cumulant generating function evaluated at a non-zero vector.

Recall that the cumulant generating function (cfg) of a p-valued random variable x is defined as

$$\phi_x(t) := \log \mathbb{E}(e^{t^\top x}), \tag{3}$$

[2] We slightly abuse the standard closely related dictionary learning terminology where the term atom is used for the individual columns d_i (see, e.g., Chen et al., 1998).

[3] The mixing component is then the largest eigenvector.

[4] The deflation part is more involved (see Section 2.4.3).

for any $t \in \mathbb{R}^p$. It is well known that the cumulants of x can be computed as the coefficients of the Taylor series expansion of the cgf evaluated at zero (see, e.g., Comon and Jutten, 2010, Chapter 5). In particular, the second order cumulant, which coincides with the covariance matrix, is then the Hessian evaluated at zero, i.e., $\mathrm{cov}(x) = \nabla^2 \phi_x(0)$.

The **generalized covariance matrix** is a straightforward extension where the Hessian of the cgf is evaluated at a non-zero vector t:

$$\mathcal{C}_x(t) := \nabla^2 \phi_x(t) = \frac{\mathbb{E}(xx^\top e^{t^\top x})}{\mathbb{E}(e^{t^\top x})} - \mathcal{E}_x(t)\mathcal{E}_x(t)^\top, \quad (4)$$

where we introduced

$$\mathcal{E}_x(t) := \nabla \phi_x(t) = \frac{\mathbb{E}(xe^{t^\top x})}{\mathbb{E}(e^{t^\top x})}. \quad (5)$$

Generalized Covariance Matrices of ICA. In case of the ICA model, substituting (1) into the expressions (5) and (4), we obtain

$$\begin{aligned} \mathcal{E}_x(t) &= \frac{D\mathbb{E}(\alpha e^{\alpha^\top y})}{\mathbb{E}(e^{\alpha^\top y})} = D\mathcal{E}_\alpha(y), \\ \mathcal{C}_x(t) &= D\mathcal{C}_\alpha(y)D^\top, \end{aligned} \quad (6)$$

where we introduced $y := D^\top t$ and the generalized covariance $\mathcal{C}_\alpha(y) := \nabla^2 \phi_\alpha(y)$ of the sources:

$$\mathcal{C}_\alpha(y) = \frac{\mathbb{E}(\alpha\alpha^\top e^{\alpha^\top y})}{\mathbb{E}(e^{\alpha^\top y})} - \mathcal{E}_\alpha(y)\mathcal{E}_\alpha(y)^\top, \quad (7)$$

where $\mathcal{E}_\alpha(y) := \nabla \phi_\alpha(y) = \mathbb{E}(\alpha e^{y^\top \alpha})/\mathbb{E}(e^{y^\top \alpha})$.

Importantly, the generalized covariance $\mathcal{C}_\alpha(y)$ of the sources, due to the independence, is a diagonal matrix (see, e.g., Podosinnikova et al., 2016). Therefore, the ICA generalized covariance $\mathcal{C}_x(t)$ is:

$$C_x(t) = \sum_{i=1}^{k} \omega_i(t) d_i d_i^\top, \quad (8)$$

where $\omega_i(t) := [\mathcal{C}_\alpha(D^\top t)]_{ii}$ are the generalized variance of the i-th source α_i. This implies that *ICA generalized covariances belong to the subspace W.*

Construction of the Subspace. Since ICA generalized covariance matrices belong to the subspace W, then the span of any number of such matrices would either be a subset of W or equal to W. Choosing sufficiently large number $s > k$ of generalized covariance matrices, we can ensure the equality. Therefore, given a sufficiently large number s of vectors $t_1, \dots, t_s$, we construct matrices $H_j := \mathcal{C}_x(t_j)$ for all $j \in [s]$. Note that in practice it is more convenient to work with vectorizations of these matrices and then consequent matricization of the obtained result (see Appendices A.1 and B.2). Given matrices H_j, for $j \in [s]$, an orthonormal basis can be straightforwardly extracted via the singular value decomposition. In practice, we set s as a multiple of k and sample the vectors t_j from the Gaussian distribution.

Note that one can also construct a basis of the subspace W from the column space of the flattening of the fourth-order cumulant of the ICA model (1). In particular, this flattening is a matrix $C \in \mathbb{R}^{p^2 \times p^2}$ such that $C = (D \odot D)\mathrm{Diag}(\kappa)(D \odot D)$, where $\odot$ stands for the Khatri-Rao product and the i-th element of the vector $\kappa \in \mathbb{R}^k$ is the kurtosis of the i-th source α_i. Importantly, matricization of the i-th column a_i of the matrix $A := D \odot D$ is exactly the i-th atom, i.e., $\mathrm{mat}(a_i) = d_i d_i^\top$. Therefore, one can construct the desirable basis from the column space of the matrix A (see Appendix B.2 for more details). This also intuitively explains the need for Assumption 2.2, which basically ensures that A has full column rank (as opposed to D). In general, this approach is common in the overcomplete literature (see, e.g., De Lathauwer et al., 2007; Bhaskara et al., 2014a; Anandkumar et al., 2015; Ma et al., 2016) and can be used as the first step of our algorithm. However, the generalized covariance-based construction has better computational (see Section 3.3) and sample complexities.

2.4 Step II: Estimation of the Atoms

We now discuss the recovery of one atom $d_i d_i^\top$, for some $i \in [k]$, given a basis of the subspace W (Section 2.4.1). We then provide a deflation procedure to recover all atoms $d_i d_i^\top$ (Section 2.4.3).

2.4.1 The Semi-Definite Program

Given matrices $H_1, H_2, \dots, H_s$ which span the subspace W defined in (2) we formulate the following *semi-definite program (SDP)*:

$$\begin{aligned} B^*_{sdp} := \operatorname*{argmax}_{B \in \mathcal{S}_p} \ & \langle G, B \rangle \\ & B \in \mathrm{Span}\{H_1, H_2, \dots, H_s\}, \\ & \mathrm{Tr}(B) = 1, \\ & B \succeq 0. \end{aligned} \quad (9)$$

We expect that the optimal solution (if it exists and is unique) B^*_{sdp} coincides with one of the atoms $d_i d_i^\top$ for some $i \in [k]$. This is not always the case, but we conjecture based on the experimental evidence that one of the atoms is recovered with high probability when $k \leq p^2/4$ (see Figure 1) and prove a weaker result (Theorem 2.1). The matrix $G \in \mathbb{R}^{p \times p}$ determines which of the atoms $d_i d_i^\top$ is the optimizer and its choice is discussed when we construct a deflation procedure (Section 2.4.3; see also Appendix C.1.4).

Intuition. Since generalized covariances $H_1, \dots, H_s$ span the subspace W, the constraint set of (9)

is:

$$\mathcal{K} := \{B \in W : \mathrm{Tr}(B) = 1, B \succeq 0\}. \quad (10)$$

It is not difficult to show (see Appendix C.2.2) that under Assumption 2.2 the atoms $d_i d_i^\top$ are extreme points of this set $\mathcal{K}$:

Lemma 2.4.1. *Let the atoms $d_1 d_1^\top$, $d_2 d_2^\top$, ..., $d_k d_k^\top$ be linearly independent. Then they are extreme points of the set $\mathcal{K}$ defined in (10).*

If the program (9) has a unique solution, the optimizer B^*_{sdp} must be an extreme point due to the compactness of the convex set $\mathcal{K}$. If the set (10) does not have other extreme points except for the atoms $d_i d_i^\top$, $i \in [k]$, then the optimizer is guaranteed to be one of the atoms. This might not be the case if the set $\mathcal{K}$ contains extreme points different from the atoms. This might explain why the phase transition (at the rate $k \leq p^2/4$) happens and could potentially be related to the phenomenon of polyhedrality of spectrahedra[5] (Bhardwaj et al., 2015).

Before diving into the analysis of this SDP, let us present its convex relaxation which enjoys certain desirable properties.

2.4.2 The Convex Relaxation

Let us rewrite (9) in an equivalent form. The constraint $B \in W := \mathrm{Span}\left\{d_1 d_1^\top, \ldots, d_k d_k^\top\right\}$ is equivalent to the fact that B is orthogonal to any matrix from the orthogonal complement (null space) of W. Let the matrices $\{F_1, F_2, \ldots, F_{m-k}\}$, where $m := p(p+1)/2$, form a basis of the null space $\mathcal{N}(W)$.[6] Then the program (9) takes an equivalent formulation:

$$\begin{aligned} B^*_{sdp} := \underset{B \in \mathcal{S}_p}{\mathrm{argmax}} \; & \langle G, B\rangle \\ & \langle B, F_j\rangle = 0, \quad \text{for all} \quad j \in [m-k], \\ & \mathrm{Tr}(B) = 1, \\ & B \succeq 0. \end{aligned} \quad (11)$$

In the presence of (e.g., finite sample) noise, the subspace W can only be estimated approximately (in the first step). Therefore, rather than keeping the hard first constraint, we introduce the relaxation

$$B^* := \underset{B \in \mathcal{S}_p}{\mathrm{argmax}} \; \langle G, B\rangle - \frac{\mu}{2} \sum_{j \in [m-k]} \langle B, F_j\rangle^2 \quad (12)$$
$$\mathrm{Tr}(B) = 1, \; B \succeq 0,$$

where $\mu > 0$ is a regularization parameter which helps to adjust to an expected level of noise. Importantly, the relaxation (12) can be solved efficiently, e.g., via the fast iterative shrinkage-thresholding algorithm (FISTA; Beck and Teboulle, 2009) and the majorization-maximization principle (see, e.g., Hunter and Lange, 2004). See Appendix C.1 for details.

2.4.3 Deflation

The semi-definite program (9), or its relaxation (12), is designed to estimate only some one atom $d_i d_i^\top$. To estimate all other atoms we need a deflation procedure. In general, there is no easy and straightforward way to perform deflation in the overcomplete setting, but we discuss possible approaches below.

Clustering. Since the matrix G determines which atom is found, it is natural to repeatedly resample this matrix a multiple of k times and then cluster the obtained atoms into k clusters. This approach generally works well except in the cases where either (a) some of the atoms, say $d_i d_i^\top$ and $d_j d_j^\top$, are relatively close (e.g., in terms of angle in the space of all symmetric matrices) to each other, or (b) one or several atoms were not properly estimated. In the former case, one could increase the number of times G is resampled, and the program is solved, but that might require very high number of repetitions. The latter issue is more difficult to fix since a single wrong atom could significantly perturb the overall outcome.

Adaptive Deflation. Alternatively, one could adapt the constraint set iteratively to exclude from the search all the atoms found so far. For that, one can update the constraint set so that the subspace W is replaced with the subspace that is spanned by all the atoms except for the ones which were already found. The most natural way to implement this is to add the found atoms to a basis of the null space of W, which is straightforward to implement with the relaxation (12). Similar to other deflation approaches, a poor estimate of an atom obtained in an earlier deflation step of such adaptive deflation can propagate this error leading to an overall poor result.

Semi-Adaptive Deflation. We found that taking advantage of both presented deflation approaches leads to the best result in practice. In particular, we combine these approaches by first performing clustering and keeping only good clusters (with low variance over the cluster) and then continuing with the adaptive deflation approach. We assume this **semi-adaptive deflation** approach for all the experiments presented in Section 3.

[5] The spectrahedron is a set formed by an intersection of the positive semi-definite cone with linear constraints, e.g. the set $\mathcal{K}$. Importantly, all polyhedra are spectrahedra, but not all spectrahedra are polyhedra.

[6] Note that a basis of $\mathcal{N}(W)$ can be easily computed given matrices $H_1, \ldots, H_s$.

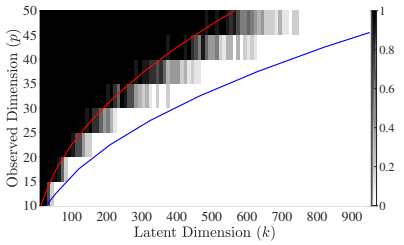

Figure 1: Phase transition of the program (13).

2.4.4 Identifiability

In general, there are two types of identifiability of probabilistic models: (a) statistical and (b) algebraic. The *statistical* identifiability addresses whether the parameters of the model can be identified for given distributions. In particular, it is well known that the ICA model is not identifiable if (more than one of) the sources are Gaussian (Comon, 1994) and issues also arise when the sources are close to Gaussian (Sokol et al., 2014). These results also extend to the overcomplete case that we consider. However, we do not address these questions here and assume that the models we work with are statistically identifiable. Instead, we are interested whether our approach is *algebraically* identifiable, i.e., whether our algorithm correctly recovers the parameters of the model. In particular, we address the following question: *When is the solution B^*_{sdp} of the program (9) is one of the atoms $d_i d_i^\top$, $i \in [k]$?*

We address this question in theory and in practice and focus on the population (infinite number of samples) case, where we assume that an exact estimate of the subspace W is given and, therefore, one can use the representation $W := \text{Span}\left\{d_1 d_1^\top, \dots, d_k d_k^\top\right\}$ without loss of generality. Therefore, for the theoretical analysis purposes we assume that atoms $d_i d_i^\top$ are known, we consider the following program instead

$$\begin{aligned} B^*_{sdp} := \underset{B \in \mathcal{S}_p}{\text{argmax}} \ & \langle G, B \rangle \\ & B \in \text{Span}\left\{d_1 d_1^\top, d_2 d_2^\top, \dots, d_k d_k^\top\right\}, \\ & \text{Tr}(B) = 1, \\ & B \succeq 0. \end{aligned} \tag{13}$$

Phase Transition. In Figure 1, we present the phase transition plot for the program (13) obtained by solving the program multiple times for different settings. In particular, for every pair (p, k) we solve the program $n_{rep} := 50$ times and assign to the respective point the value equal to the fraction of successful solutions (where the optimizer was one of the atoms).

Given a fixed pair (p, k), every instance of the program (13) is constructed as follows. We first sample a mixing matrix $D \in \mathbb{R}^{p \times k}$ so that every mixing component is from the standard normal distribution as described in Appendix D.1; and we sample a matrix $G \in \mathbb{R}^{p \times p}$ from the standard normal distribution. We then construct the constraint set of the program (13) by setting every matrix $H_i = d_i d_i^\top$ for all $i \in [k]$, where $s = k$. We solve every instance of this problem with the CVX toolbox (Grant et al., 2006) using the SeDuMi solver (Sturm, 1999).

We consider the observations dimensions p from 10 to 50 with the interval of 5 and we vary the number of atoms from 10 to 1000 with the interval of 10. The resulting phase transition plots are presented in Figure 1. The **blue line** on this plot corresponds to the curve $k = p(p+1)/2$, which is the largest possible latent dimension of all symmetric matrices $\mathcal{S}_p$. The **red line** on this plot corresponds to the curve $k = p^2/4$. Since above the red line we observe 100% successful recovery (black), *we conjecture that the phase transition happens around $k = p^2/4$.*

Theoretical Results. Interestingly, an equivalent conjecture, $k < p^2/4$, was made for the ellipsoid fitting problem (Saunderson et al., 2012, 2013) and the question remains open to our best knowledge.[7] In fact, we show close relation between successful solution (recovery of an atom) of our program (13) and the ellipsoid fitting problem. In particular, a successful solution of our problem implies that the feasibility of its Lagrange dual program is equivalent to the ellipsoid fitting problem (see Appendix C.2.3). Moreover, using this connection, we prove the following:

Theorem 2.1. *Let $\varepsilon > 0$. Consider a regime with p tending to to infinity, and with k varying according to the bound $k < (2-\varepsilon) p \log p$. As above, let the d_i be random unit vectors and let $G = uu^\top$ for a random unit vector u. Then with high probability[8], the matrix $d_i d_i^\top$ for which $d_i^\top G d_i$ is largest is the unique maximizer of the program (13).*

3 Experiments

It is difficult to objectively evaluate unsupervised learning algorithms on real data in the absence of ground truth parameters. Therefore, we first perform comparison on synthetic data. All our experiments can be reproduced with the publicly available code: `https://github.com/anastasia-podosinnikova/oica`.

[7] In Appendix C.2.1, we recall the formulation of the ellipsoid fitting problem and slightly improve the results of Saunderson et al. (2012, 2013).

[8] Throughout, "with high probability" indicates probability tending to 1 as $p \to \infty$.

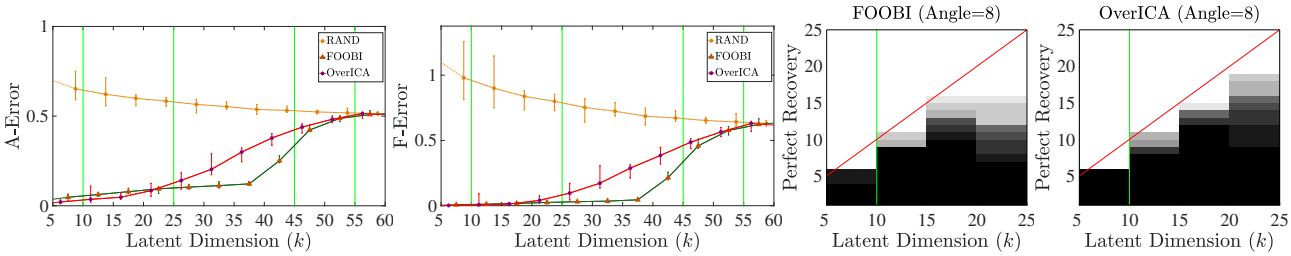

Figure 2: A proof of concept in the asymptotic regime. See explanation in Section 3.1.

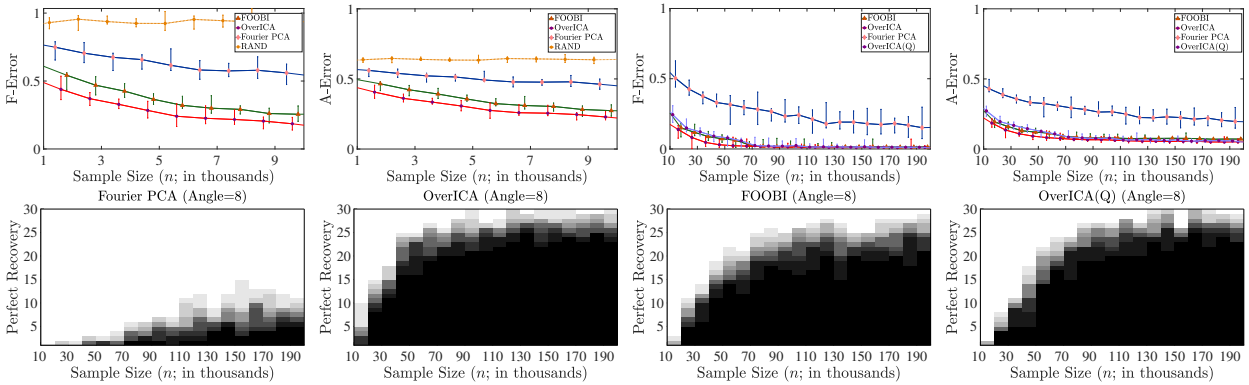

Figure 3: Comparison in the finite sample regime. See explanation in Section 3.2.

3.1 Synthetic Data: Population Case

As a proof of concept, this simple experiment (Figure 2) imitates the infinite sample case. Given a ground truth mixing matrix D, we construct a basis of the subspace W directly from the matrix $A := D \odot D$ (see Appendix B.2). This leads to a noiseless estimate of the subspace. We then evaluate the performance of the second step of our OverICA algorithm and compare it with the second step of FOOBI. We fix the observed dimension $p = 10$ and vary the latent dimension from $k = 5$ to $k = 60$ in steps of 5. For every pair (p, k), we repeat the experiment $n_{rep} = 10$ times and display the minimum, median, and maximum values. Each time we sample the mixing matrix D with mixing components from the standard normal distribution (see Appendix D.1.1). Note that we tried different sampling methods and distributions of the mixing components, but did not observe any significant difference in the overall result. See Appendix D.1.2 for further details on this sampling procedure.

The error metrics (formally defined in Appendix D.2) are: (a) f-error is essentially the relative Frobenius norm of the mixing matrices with properly permuted mixing components (lower is better); (b) a-error measures the angle deviations of the estimated mixing components vs the ground truth (lower is better); and (c) "perfect" recovery rates, which show for every $i \in [k]$ the fraction of perfectly estimated i components. We say that a mixing component is "perfectly" recovered if the cosine of the angle between this component d_i and its ground truth equivalent $d_{\pi(i)}$ is at least 0.99, i.e., $\cos(d_i, \widehat{d}_{\pi(i)}) \geq 0.99$. Note that the respective angle is approximately equal to 8. Then the black-and-white perfect recovery plots (in Figure 2) show if $i \leq k$ (on the y-axis) components were perfectly recovered (black) for the given latent dimension k (x-axis). These black vertical bars cannot exceed the red line $i = k$, but the closer they approach this line, the better. The vertical green lines correspond to $k = p = 10$, $k = p^2/4 = 25$, $k = p(p-1)/2$, and $k = p(p+1)/2$. Importantly, we see that OverICA works better or comparably to FOOBI in the regime $k < p^2/4$. Performance of OverICA starts to deteriorate near the regime $k \approx p^2/4$ and beyond, which is in accord with our theoretical results in Section 2.4.4. Note that to see whether the algorithms work better than random, we display the errors of a randomly sampled mixing matrix (RAND; see Appendix D.1.1).

3.2 Synthetic Data: Finite Sample Case

With these synthetic data we evaluate performance of overcomplete ICA algorithms in the presence of finite sample noise but absence of model misspecification. In particular, we sample synthetic data in the observed dimension $p = 15$ from the ICA model with uniformly distributed (on $[-0.5, 0.5]$) $k = 30$ sources for different sample sizes n taking values from $n = 1,000$ to $n = 10,000$ in steps of $1,000$ (two left most plots in the top line of Figure 3) and values from $n = 10,000$ to $n = 210,000$ in steps of $10,000$ (two right most plots in

Table 1: Computational complexities (n is the sample size, p is the observed dimension, k is the latent dimension, s is the number of generalized covariances, usually $s = O(k)$).

Procedure	Memory	Time
GenCov	$O(p^2 s)$	$O(snp^2)$
CUM	$O(p^4)$	$O(np^4 + k^2p^2)$
FOOBI	$O(p^4k^2 + k^4)$	$O(np^4 + k^2p^4 + k^6)$
OverICA	$O(sp^2)$	$O(nsp^2)$
OverICA(Q)	$O(p^4)$	$O(np^4 + k^2p^2)$
Fourier PCA	$O(p^4)$	$O(np^4)$

the top line of Figure 3; see also Figure 6 in Appendix for log-linear scale). Note that the choice of dimensions $p = 15$ and $k = 30$ corresponds to the regime $k < p^2/4 \approx 56$ of our guarantees. We repeat the experiment $n_{rep} := 10$ times for every n where we every time resample the (ground truth) mixing matrix (with the sampling procedure described in Appendix D.1.1). See further explanation in Appendix D.1.3.

We compare the Fourier PCA algorithm (Goyal et al., 2014), the FOOBI algorithm (De Lathauwer et al., 2007), OverICA from Algorithm 1, and a version of the OverICA algorithm where the first step is replaced with the construction based on the fourth-order cumulant, a.k.a. quadricovariance (OverICA(Q); see Appendix B.2). Note that we can not compare with the reconstruction ICA algorithm by Le et al. (2011) because it estimates the de-mixing (instead of mixing) matrix.[9] Similarly to Section 3.1, we measure the Frobenius error (f-error), the angle error (a-error), and the perfect recovery for the angle of 8. We observe that the generalized covariance-based OverICA algorithm performs slightly better which we believe is due to the lower sample complexity. Fourier PCA on the contrary performs with larger error, which is probably due to the higher sample complexity and larger noise resulting from estimation using fourth-order generalized cumulants.

3.3 Computational Complexities

In Table 1, we summarize the timespace complexities of the considered overcomplete ICA algorithms and two sub-procedures they use: generalized covariances (GenCov; used by OverICA) from Section 2.3 and the forth-order cumulant (CUM; used by OverICA(Q) and FOOBI; see Appendix B.2) (see Appendix D.3). Importantly, we can see that our OverICA algorithm has a significantly lower complexity. In Appendix D.3, we present runtime comparisons of these algorithms.

[9] In the complete invertible case, the *de-mixing matrix* would be the inverse of the mixing matrix. In the overcomplete regime, one cannot simply obtain the mixing matrix from the de-mixing matrix.

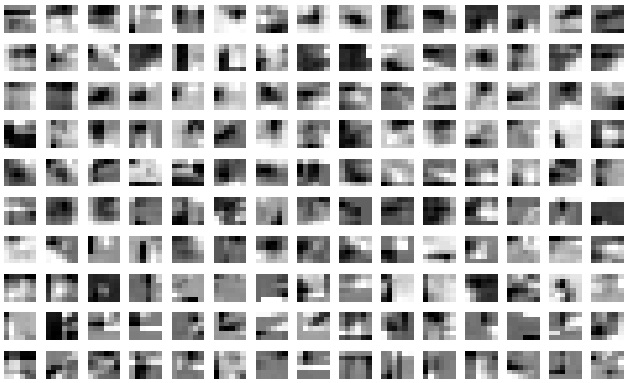

Figure 4: Mixing components obtained from 7-by-7 patches, i.e., $p = 49$, of the CIFAR-10 dataset ($k = 150$, i.e., overcomplete). ICA does not preserve non-negativity and the signs of ICA mixing components can be arbitrarily flipped due to the scaling unidentifiability; here black and white correspond to the extreme positive and extreme negative values. The colorbar limits of every image are the same and the signs are alligned to have positive scalar product with the first component.

3.4 Real Data: CIFAR-10 Patches

Finally, we estimate the overcomplete mixing matrix of data formed of patches of the CIFAR-10 dataset (see, e.g., Krizhevsky et al., 2014). In particular, we transform the images into greyscale and then form 7-by-7 patches for every interior point (at least 3 pixels from the boundary) of every image from the training batch 1 of the CIFAR-10 dataset. This results in $6,760,000$ patches each of dimension $p = 49$. We perform the estimation of the mixing matrix for $k = 150$ latent mixing components. The resulting atoms are presented in Figure 4. Note that since ICA is scale (and therefore sign) invariant, the sign of every component can be arbitrary flipped. We present the obtained components in the scale where black and white corresponds to the extreme positive or negative values and we observe that these peaks are concentrated in rather pointed areas (which is a desirable property of latent components). Note that the runtime of this whole procedure was around 2 hours on a laptop. Due to high timespace complexities (see Section 3.3), we cannot perform similar estimation neither with FOOBI nor with Fourier PCA algorithms.

4 Conclusion

We presented a novel ICA algorithm for estimation of the latent overcomplete mixing matrix. Our algorithm also works in the (under-)complete setting, enjoys lower computational complexity, and comes with theoretical guarantees, which is also confirmed by experiments.

Acknowledgements

A. Podosinnikova was partially supported by DARPA grant #W911NF-16-1-0551. A. Podosinnikova and D. Sontag were partially supported by NSF CAREER award #1350965. This work was supported in part by NSF CAREER Award CCF-1453261 and a grant from the MIT NEC Corporation. Part of this work was done while A. S. Wein was at the Massachusetts Institute of Technology. A. S. Wein received Government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a. A. S. Wein is also supported by NSF grant DMS-1712730 and by the Simons Collaboration on Algorithms and Geometry.

References

A. Anandkumar, R. Ge, and M. Janzamin. Learning overcomplete latent variable models through tensor methods. In *Proceedings of the Conference on Learning Theory (COLT)*, 2015.

S. Arora, R. Ge, A. Moitra, and S. Sachdeva. Provable ICA with unknown Gaussian noise, with implications for Gaussian mixtures and autoencoders. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

A. Bhardwaj, P. Rostalski, and R. Sanyal. Deciding polyhedrality of spectrahedra. *SIAM Journal on Optimization*, 25(3):1873–1884, 2015.

A. Bhaskara, M. Charikar, A. Moitra, and A. Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, 2014a.

A. Bhaskara, M. Charikar, and A. Vijayaraghavan. Uniqueness of tensor decompositions with applications to polynomial idenfifiability. In *Proceedings of the Conference on Learning Theorey (COLT)*, 2014b.

A. Bovier. Extreme Values of Random Processes. *Lecture Notes Technische Universität Berlin*, 2005.

A. Bunse-Gerstner, R. Byers, and V. Mehrmann. Numerical methods for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications*, 14(4):927–949, 1993.

J.-F. Cardoso and A. Souloumiac. Blind beamforming for non-Gaussian signals. In *IEE Proceedings F - Radar and Signal Processing*. IEEE, 1993.

J.-F. Cardoso and A. Souloumiac. Jacobi angles for simultaneous diagonalization. *SIAM Journal on Matrix Analysis and Applications*, 17(1):161–164, 1996.

S.S. Chen and D.L. Donoho. Basis Pursuit. Technical report, Stanford University, 1994.

S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

A. Coates, H. Lee, and A.Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

P. Comon. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314, 1994.

P. Comon and C. Jutten. *Handbook of Blind Sourse Separation: Independent Component Analysis and Applications*. Academic Press, 2010.

P. Comon and M. Rajih. Blind identification of under-determined mixtures based on the characteristic function. *Signal Processing*, 86(9):2271–2281, 2006.

I. Daubechies. Time-frequency localization operators: A geometric phase space approach. *IEEE Transactions on Information Theory*, 34(4):604–612, 1988.

L. De Lathauwer, J. Castaing, and J.-F. Cardoso. Fourth-order cumulant-based blind identification of underdetermined mixtures. *IEEE Transactions on Signal Processing*, 55(6):2965–2973, 2007.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2008.

N. Goyal, S. Vempala, and Y. Xiao. Fourier PCA and robust tensor decomposition. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, 2014.

M. Grant, S. Boyd, and Y. Ye. Disciplined convex programming. In *Global Optimization: from Theory to Implementation, Nonconvex Optimization and Its Applications*. Springer, 2006.

D.R. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research (JMLR)*, 6:695–708, 2005.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001.

A. Krizhevsky, V. Nair, and G. Hinton. The CIFAR-10 dataset. *University of Toronto*, 2014. URL http://www.cs.toronto.edu/kriz/cifar.html.

H.W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

Q.L. Le, A. Karpenko, J. Ngiam, and A.Y. Ng. ICA with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

M.S. Lewicki and T.J Sejnowski. Learning overcomplete representations. *Neural Computation*, 12(2): 337–365, 2000.

T. Ma, J. Shi, and D. Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *Annual Symposium on Foundations of Computer Science (FOCS)*, 2016.

B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

B.A. Olshausen and D.J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

A. Podosinnikova, F. Bach, and S. Lacoste-Julien. Beyond CCA: Moment matching for multi-view models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

J. Saunderson. *Subspace identification via convex optimization*. PhD thesis, Massachusetts Institute of Technology, 2011.

J. Saunderson, V. Chandrasekaran, P.A. Parrilo, and A.S. Willsky. Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting. Technical report, arXiv:1204.1220v1, 2012.

J. Saunderson, P.A. Parrilo, and A.S. Willsky. Diagonal and low-rank decompositions and fitting ellipsoids to random points. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 2013.

A. Sokol, M.H. Maathuis, and B. Falkeborg. Quantifying identifiability in independent component analysis. *Electronic Journal of Statistics*, 8:1438–1459, 2014.

J.F. Sturm. Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(12):625–633, 1999.

Y.W. Teh, M. Welling, S. Osindero, and G.E. Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research (JMLR)*, 4:1235–1260, 2003.

A. Yeredor. Blind source separation via the second characteristic function. *Signal Processing*, 80(5): 897–902, 2000.