Transfer Learning-based Independent Component Analysis

Ziqian Zheng, Wei Zhao, Brock Hable, Yutao Gong, Xuan Wang, Robert W. Shannon, and Kaibo Liu, *Senior Member, IEEE*

***Abstract*—Understanding the underlying component structure is crucial for multivariate signal analysis. Among all the techniques that try to learn the latent structure, independent component analysis (ICA) is one of the most important and popular methods, which aims to extract independent components from multivariate signals and enables further analysis. For example, in electroencephalogram (EEG) analysis, artifacts filtering and disease detection are conducted based on the independent components of the signals. One critical challenge in existing ICA approaches is that the component extraction accuracy may degrade when the available data of a unit are limited. To address this issue, this paper proposes a transfer learning-based ICA method by innovatively transferring component distribution from a source domain, so that accurate component extraction results can be achieved even when only limited data are available in the target domain. To the best of our knowledge, this is the first work that leverages transfer learning to improve ICA accuracy with limited available data. In particular, we first extract all the independent components from the source domain by maximizing the log-likelihood function with a Newton-like method on a smooth manifold. Then for the target domain, the component with the largest negentropy is extracted in each round. To effectively leverage the knowledge from the source domain and to prevent the negative transfer, we try to find a component in the source domain that matches the component we are extracting. The probability density function of the matched component will then be used to improve the component extraction accuracy if such matched component can be found; otherwise, no knowledge will be transferred. Numerical simulations and a case study with electrocardiogram (ECG) data are conducted, showing the effectiveness of the proposed method in transferring knowledge and reducing negative transfer.**

***Note to Practitioners*—This paper is motivated by the practical issue of conducting independent component analysis when data are limited to derive a reliable result. To address this challenge, we propose to transfer knowledge from a source domain to the target domain. Specifically, there are two fundamental questions involved: 1) what knowledge can be transferred from the source domain; and 2) how to minimize the negative transfer when no useful knowledge is available. Our novel idea is to transfer the component distributions and the negative transfer is largely reduced through a component matching step as a result. There are four main steps involved when implementing the proposed method: 1) solve all the independent components in the source domain; 2) extract the independent component in the target domain with the largest negentropy; 3) decide whether a matched component can be found from the source domain; and 4) re-estimate the current independent component in the target domain by leveraging the distribution information of the matched component. Step 2 to step 4 are repeated several times until all the independent components in the target domain are extracted or some termination condition is reached.**

***Index Terms*—ICA, independent component matching, limited data, negative transfer.**

I. INTRODUCTION

With recent advances in sensor technology, multivariate signals have been widely used in all kinds of areas such as healthcare and manufacturing. For example, in manufacturing, multivariate signals are used for degradation modeling to avoid unexpected failures [1][2]; in healthcare, multivariate signals such as EEG, ECG, and functional magnetic resonance imaging (fMRI) provide a wealth of information about brain and heart function and by analyzing these signals, potential diseases can be detected at a very early stage and treated effectively [3]-[5]. In reality, it is common that signal streams collected by different sensors are correlated and each signal stream only contains partial information about the true system status. Therefore, how to accurately learn the underlying structure of the multivariate signals is crucial for successful data analysis and further decision-making.

Among all the techniques of evaluating the underlying structure involved in multivariate signals, ICA is one of the most widely used methods, due to its reasonable assumptions and nice interpretation in practice. Generally speaking, ICA aims to separate multivariate signals into statistically independent components, and each signal is assumed to be a linear combination of some independent components. Such assumption is reasonable in practical applications, as the collected sensor observations are often combinations of latent independent processes with some noise. Specifically, the typical form of ICA is as follows,

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}, \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^m$ is a vector observation of m sensors, $\boldsymbol{A} \in \mathbb{R}^{m \times k}$

This work was supported in part by 3M and in part by the Department of Energy under award number DE-NE0008993 (*Corresponding author: Kaibo Liu.*)

Z. Zheng and K. Liu are with the Department of Industrial and Systems Engineering, University of Wisconsin-Madison, Madison, WI 53706 USA (e-mail: zzheng92@wisc.edu; kliu8@wisc.edu).

W. Zhao, B. Hable, Y. Gong, J. Wang, and R. W. Shannon, are with the 3M Company, Maplewood, MN USA (e-mail: wzhao5@mmm.com, bahable@mmm.com, ygong4@mmm.com, xwang15@mmm.com, rshannon@mmm.com).

is the mixing matrix, $\boldsymbol{s} \in \mathbb{R}^k$ is a vector sample of k independent components, and $\boldsymbol{e} \in \mathbb{R}^m$ denotes the random noise. The goal of ICA is to estimate the mixing matrix and independent components based on a collection of observed sensor measurements.

Although ICA has achieved great success in dealing with multivariate signals in many applications, existing literature assumes that there are enough data available to conduct the ICA. However, this assumption may not always be valid. For example, let us consider a new machine working in the production plant, or a healthy person who only conducts the ECG test for a limited time during a regular physical examination. In both cases, the multivariate data can be sparse and the conventional ICA algorithms based on the limited data may result in a very large error.

To address this limitation, we propose to improve the ICA estimation accuracy when the available data are limited by transfer learning. More specifically, transfer learning transfers knowledge from a source domain to the target domain and aims to improve the performance of a learning task (i.e., ICA) in the target domain. However, two problems remain unsolved:

(1) *What knowledge can be transferred.* Deciding on the knowledge that can be transferred is one of the most important tasks in transfer learning and will directly influence the performance in the target domain. The transferred knowledge should be able to represent the commonalities between the two domains.

(2) *How to prevent negative transfer*. Negative transfer means that leveraging knowledge from the source domain reduces the performance in the target domain. This usually happens when the source domain and the target domain are very different, and no commonalities are shared. Consequently, it is desired that the proposed method can reduce the negative transfer by transferring minimum knowledge or no knowledge when no useful information from the source domain is available.

To resolve these two problems, this paper proposes a novel transfer learning idea, called component **M**atching-based **ICA** (denoted as "MICA"), that transfers the component distribution information from the source domain and improves the ICA accuracy in the target domain. In particular, knowledge is transferred only if we detect commonalities that are shared by both domains through a component matching step.

Consequently, the contribution of this paper is summarized as follows. First, unlike the existing transfer learning methods that often transfer model parameters, data features, or instances, the transferred knowledge in the proposed method, which is the distribution information, is specifically tailored for the characteristics of ICA. To the best of our knowledge, this is the first work that leverages transfer learning to improve ICA accuracy when only limited data are available. Second, we only assume that some components from different domains are matched (i.e., they have the same distribution) and no assumptions are made to the mixing matrices in the source and target domains. Thus, the proposed method is generic and can be applied to various ICA applications with limited data. Finally, no knowledge is transferred when there is no matched component found from the source domain. This component matching step helps reduce the negative transfer when no useful knowledge is shared in the source and target domains.

The rest of this paper is organized as follows. Section II reviews the existing literature on ICA and transfer learning. Section III describes the details of the problem formulation. The proposed MICA method is introduced in section IV. Section V conducts comprehensive simulation studies to illustrate the effectiveness of the proposed method and compare it with existing benchmarks. In section VI, the proposed method is further tested using an ECG dataset. Finally, Section VII provides a conclusion and a discussion of future research work.

II. Literature Review

This section contains two subsections. In Section II.A, we review the methodologies of ICA. Then in Section II.B, we briefly introduce the transfer learning approach.

A. Independent Component Analysis

ICA is a statistical method for separating multivariate signals into statistically independent components and has been successfully applied to a lot of applications, such as EEG analysis [5], fMRI analysis [4], process monitoring [6], human transcriptome analysis [7], and price forecasting [8].

The fundamental idea of solving the ICA is to maximize the statistical independence of the separated components [9][10]. Following this idea, lots of methods have been proposed by introducing different independence criteria and they can be classified into two main categories: parametric approach and nonparametric approach.

For the parametric approach, *Hyvärinen and Oja* [9] proposed the *FastICA* algorithm that maximizes the component independence by maximizing the negentropy of the component, where the negentropy is estimated with certain functions. *Cardoso* [11] used cumulants as measures of independence and extracted the components by minimizing the cumulants. One common drawback of these parametric methods is that higher-order moments used in these methods are vulnerable to outliers in the dataset. Besides, some additional assumptions must be made in the parametric approaches and these methods can fail when such assumptions are not satisfied.

To overcome the disadvantages above, several efforts have been made to construct nonparametric independence criteria. For example, some methods [13]-[15] maximized the likelihood function based on the nonparametric modeling of the probability density functions. Other nonparametric methods derived the independence criteria by mapping the data to the feature spaces (i.e., reproducing kernel Hilbert spaces) and considering the spectrum of the covariance operator. These kernel independence criteria include the Hilbert-Schmidt independence criterion [16], kernel mutual information [17], and kernel generalized variance [18].

Though the reviewed methods above can achieve good performance in a wide range of applications, one common limitation is that none of them considered the issue of data

sparsity. To fill the literature gap, in this paper, we borrow the idea from transfer learning to improve the ICA performance when only limited data are available.

B. Transfer Learning

In recent years, transfer learning has emerged as a very popular approach to tackle data sparsity. The core idea of transfer learning is transferring useful knowledge from one data-rich domain (called source domain) and improving the performance in the target domain that is often associated with limited data. Depending on what knowledge to be transferred, transfer learning can be mainly categorized into three types [19]: instance-transfer, feature-transfer, and parameter-transfer.

The instance-transfer approach augments the target domain data by transferring data from the source domain. For example, a boosting algorithm called *TrAdaBoost* was proposed in [20], which iteratively reweighted the source domain data and added them to the target domain for the classification task. Later, Yao and Doretto [21] extended the *TrAdaBoost* algorithm for transferring knowledge from multiple sources and proposed two new algorithms: *MultiSource-TrAdaBoost* and *TaskTrAdaBoost*. Though these boosting algorithms are efficient in classification or regression tasks, such instance-transfer approach cannot be applied to ICA. The reason is that the data in the two domains may have different dimensions, and thus we cannot directly augment the target domain data with the data from the source domain. In addition, the mixing matrices in different domains can be very different as well, making the instance-transfer approach even unjustifiable.

Feature-transfer approach aims to learn a common feature representation that can reduce the difference between the source and the target domains. Pan *et. al.* [22] proposed a transfer learning method called transfer component analysis (TCA) that can learn a domain-invariant feature representation to minimize the marginal distribution difference between the source and the target domains. Long *et. al.* [23] extended the TCA by matching the marginal distribution and conditional distribution between the two domains simultaneously. More recently, the feature-transfer approach has also been adopted to boost the performance of the deep neural network [24]-[26]. One common limitation is that all these works use the same transformation for the source and target domains to extract features. This approach might be reasonable for some tasks like computer vision. However, if the mixing matrices are different in different domains, we may not be able to find a single transformation that extracts the independent components from the source and target domains simultaneously.

Finally, parameter-transfer approach tries to find the shared model parameters or prior distributions between the models in the two domains. One of the most widely used parameter-transfer approaches is the fine-tuning of the neural network, which borrows the network weights from the network that is already trained in the source domain and only updates the last few layers based on the data in the target domain [27]. Although fine-tuning is tested to be useful in a lot of applications [28][29], it is not proper for our problem. One reason is that the fine-tuning method assumes the first few layers are good enough to extract the same features from both source domain data and target domain data. Again, this assumption may not be valid if the mixing matrices are different in the source and target domains, since different calculations should be conducted to extract independent components from different domains. Besides, by using fine-tuning, we usually need a very large dataset in the source domain to pre-train the neural network. This condition may not hold in practice as well.

In summary, the existing literature still lacks a generic and effective transfer learning method that efficiently improves the ICA performance in the target domain with limited data.

III. Problem Formulation

Suppose we have collected data from two domains: the source domain and the target domain, which are denoted as s and t. In domain $d \in \{s,t\}$, the collected data are denoted as $\boldsymbol{X}^{(d)}$, where $\boldsymbol{X}^{(d)} \in \mathbb{R}^{m_d \times n_d}$ is a full rank matrix with rank m_d, m_d is the number of data streams and n_d is the number of observations. Without loss of generality, we assume the mean of each data stream is 0 after centering. In the context of ICA [30], each observation of $\boldsymbol{X}^{(d)}$ arises from a linear mixing of independent components plus some white noise. That is,

$$\boldsymbol{X}^{(d)} = \boldsymbol{A}^{(d)}\boldsymbol{S}^{(d)} + \boldsymbol{E}^{(d)}, d \in \{s,t\}, \quad (2)$$

where $\boldsymbol{A}^{(d)} \in \mathbb{R}^{m_d \times k_d}$ is the mixing matrix, $\boldsymbol{S}^{(d)} \in \mathbb{R}^{k_d \times n_d}$ is a matrix containing n_d realizations of k_d latent components that are sampled from certain distributions, $\boldsymbol{E}^{(d)} \in \mathbb{R}^{m_d \times n_d}$ is the noise matrix and each column of $\boldsymbol{E}^{(d)}$ independently follows a multivariate gaussian distribution with mean $\mathbf{0}$ and a diagonal covariance matrix $\boldsymbol{U}^{(d)}$, i.e., $\mathcal{N}\left(\mathbf{0}, \boldsymbol{U}^{(d)}\right)$. In (2) above, only the observations $\boldsymbol{X}^{(d)}$, $d \in \{s,t\}$, are known. Usually, the number of components n_d and the number of data streams m_d are not equal, since the latter is determined by the number of sensors in the system. Following the existing literature, we assume that $m_d \geq k_d$, and $rank(\boldsymbol{A}^{(d)}) = k_d$, for $d \in \{s,t\}$, such that the independent components are identifiable in both domains. In the rest of this paper, if not otherwise specified, let m_{ij}, $\boldsymbol{m}_{i\cdot}$ and $\boldsymbol{m}_j$ denote the ijth entry, the ith row, and the jth column of a matrix $\boldsymbol{M}$, respectively. It is also assumed that for $d \in \{s,t\}$, $i = 1,2,\dots,k_d$, $j = 1,2,\dots,n_d$, $s_{ij}^{(d)}$ follows a continuous distribution with probability density function (PDF) $f_i^{(d)}$, and $s_{ij}^{(d)}$ and $s_{i'j}^{(d)}$ are independent if $i \neq i'$. This is the reason why the method is called independent component analysis.

We further assume that the target domain is a new domain of our interest, and the number of observations is much less than that of the source domain, i.e., $n_s \gg n_t$. This happens, for example, when the target domain represents a new patient just entering the study, while the source domain represents a patient that has been monitored for a long time. Due to the lack of data in the target domain, the independent components recovering is unreliable in the target domain. In addition, we cannot combine the data from the two domains to perform ICA when the sets of sensors used or the mixing matrices in the two domains are different. Consequently, the goal of this study is to establish an appropriate transfer learning approach tailored for the ICA in

(2), such that the knowledge from the source domain can be effectively employed to accurately recover the latent components in the target domain.

When conducting ICA, there are three common issues in recovering latent independent components [9]. To make the explanation clearer, we omit the superscript for now and let $\boldsymbol{X}$, $\boldsymbol{A}$, $\boldsymbol{S}$ denote the data matrix, mixing matrix, and component matrix, respectively. The first issue is that the variance of each independent component is often undetermined. Since both $\boldsymbol{S}$ and $\boldsymbol{A}$ are unknown, if $\boldsymbol{s}_{i\cdot}$ is replaced by $\delta\boldsymbol{s}_{i\cdot}$, we can simply replace $\boldsymbol{a}_i$ by $\frac{\boldsymbol{a}_i}{\delta}$ to obtain the same data matrix $\boldsymbol{X}$, where $\boldsymbol{s}_{i\cdot}$ is the ith row of $\boldsymbol{S}$ and $\boldsymbol{a}_i$ is the jth column of $\boldsymbol{A}$. Second, the order of the independent components is undetermined as well. This is because any permutation of the rows of $\boldsymbol{S}$ could always be canceled by a permutation of columns in $\boldsymbol{A}$. Finally, only non-Gaussian independent components are separable. For the gaussian components, if they have the same variance, they can only be estimated up to an orthogonal transformation. For instance, let $\boldsymbol{s} \in \mathbb{R}^{k\times 1}$ denote a sample of k independent components and $\boldsymbol{s} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. Then for any orthogonal matrix $\boldsymbol{O} \in \mathbb{R}^{k\times k}$, $\boldsymbol{Os} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$. Therefore, the matrix $\boldsymbol{OS}$ can also be regarded as a collection of independent components, where the mixing matrix is $\boldsymbol{AO}^T$. To address these issues, following the existing literature, each independent component is assumed to have a unit variance, and we only aim to extract all non-Gaussian components up to permutation and multiplicative signs. Note that after all non-Gaussian components are extracted, we can then automatically obtain the remaining Gaussian components up to an orthogonal transformation. However, the Gaussian components are not the focus of this work.

It is worth mentioning that we do not require $m_s = m_t$ or $k_s = k_t$, and $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ can also be different. The only commonality of the two domains is that they may share some latent components, which means some components in the two domains follow the same distribution. However, there is no prior knowledge about which and how many component distributions are shared by both domains. When there is no commonality in the two domains, we should automatically transfer as little knowledge as possible to prevent negative transfer. We should note that the considered commonality is very natural in the real world. For example, each independent component of the multivariate EEG signals may have some physical meaning, such as eye blink topography, oscillations in alpha rhythm, and brain activity that is related to acoustic or visual simulation [31]. For different people, the components corresponding to the same brain activities may follow the same distributions. However, since the mixing matrices are influenced by some other factors such as the cranium shape and the location of the sensor, the mixing matrices can be different for different people. To the best of our knowledge, this is the first work that combines transfer learning and ICA to improve the component extraction accuracy when the available data in the target domain is limited.

IV. Methodology

In this section, we will introduce the proposed transfer learning-based ICA method in detail. As mentioned before, the knowledge that can be potentially transferred from the source domain to the target domain is the distribution information of source domain components. Therefore, in Section IV.A, we first discuss the extraction of the independent components in the source domain. Then, in Section IV.B, we present the proposed transfer learning method.

A. Independent component analysis in the source domain

When only considering the components extraction of the source domain, the problem reduces to the conventional ICA. In the literature, many algorithms have been proposed. Among them, *FastICA* [9] is one of the most widely used methods to solve ICA due to its simplicity. In *FastICA*, the negentropy is estimated by certain functions and the components are extracted by maximizing the negentropy. However, [13]-[15] suggested that one can achieve better accuracy by maximizing the log-likelihood function with nonparametric density estimation. More importantly, in using the likelihood function, it is much more convenient to transfer the component distributions from the source domain to the target domain. It is also well known that maximizing the log-likelihood function is equivalent to minimizing the mutual information between the estimated independent components [9], and is also equivalent to making the extracted components as independent as possible. Hence, in this study, we follow the literature to nonparametrically estimate the density function of the components and extract the components by maximizing the log-likelihood function.

Following the existing studies [9][32], to extract the independent components in the source domain, it requires to find an unmixing matrix $\boldsymbol{W}$, such that the reconstructed component matrix is as follows,

$$\widehat{\boldsymbol{S}} = \boldsymbol{WX}. \tag{3}$$

However, there is one issue that the number of independent components, k, is unknown and we only know that $k \le m$. To resolve this problem, we let $\boldsymbol{W} \in \mathbb{R}^{m\times m}$ and aim to reconstruct m components from the m data streams. Among them, k components should be the true components, and the rest $m-k$ components will be independent Gaussian noise. Existing studies naturally assume such $\boldsymbol{W}$ exists. Here, to the best of our knowledge, we first rigorously prove the existence of such an oracle $\boldsymbol{W}$ in Theorem 1.

Theorem 1: Suppose the data matrix $\boldsymbol{X}$ has full rank, then there exists an unmixing matrix $\boldsymbol{W}$, such that

$$\hat{\boldsymbol{s}}_j = \boldsymbol{Wx}_j = \begin{bmatrix} \boldsymbol{s}_j + \boldsymbol{e}_{j1} \\ \boldsymbol{e}_{j2} \end{bmatrix}, \qquad j = 1,2,\cdots,n, \tag{4}$$

where both $\boldsymbol{e}_{j1} \in \mathbb{R}^{m\times 1}$ and $\boldsymbol{e}_{j2} \in \mathbb{R}^{(m-k)\times 1}$ follow the multivariate Gaussian distribution with mean $\boldsymbol{0}$, the covariance matrix of $\boldsymbol{e}_{j2}$ is the identity matrix $\boldsymbol{I}_{m-k}$, and $\boldsymbol{s}$, $\boldsymbol{e}_{j1}$ and $\boldsymbol{e}_{j2}$ are mutually independent.

Proof: See the Appendix A.

In practice, to simplify the problem of solving such $\boldsymbol{W}$, a preprocessing step called whitening is often carried out [1]. Precisely, the covariance matrix of an observation can be

estimated as

$$\boldsymbol{\Sigma} \approx \frac{1}{n}\sum_{j}^{n} \boldsymbol{x}_j \boldsymbol{x}_j^T , \tag{5}$$

where $\boldsymbol{x}_j$ is the jth column of $\boldsymbol{X}$. Note that $\boldsymbol{\Sigma}$ is symmetry and positive definite, which implies that $\boldsymbol{\Sigma} = \boldsymbol{P}\boldsymbol{D}\boldsymbol{P}^T$, where $\boldsymbol{P}$ is an orthogonal matrix and $\boldsymbol{D}$ is a diagonal matrix with positive diagonal values $(d_1, d_2, \dots, d_m)$. Then we can define the diagonal matrix $\boldsymbol{D}^{-\frac{1}{2}} = diag\left(d_1^{-\frac{1}{2}}, d_2^{-\frac{1}{2}}, \dots, d_m^{-\frac{1}{2}}\right)$, and the whitening process can be expressed as

$$\boldsymbol{z}_j = \boldsymbol{P}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{P}^T\boldsymbol{x}_j, \qquad j = 1,2,\cdots,n, \tag{6}$$

where the covariance matrix of $\boldsymbol{z}_j$ is the identity matrix. Accordingly, one can estimate the components $\hat{\boldsymbol{s}}_j$ by a linear transformation of $\boldsymbol{z}_j$, that is

$$\hat{\boldsymbol{s}}_j = \boldsymbol{B}^T\boldsymbol{z}_j = \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{P}^T\boldsymbol{x}_j, \qquad j = 1,2,\cdots,n. \tag{7}$$

Since both $\hat{\boldsymbol{s}}_j$ and $\boldsymbol{z}_j$ have an identity covariance matrix, $\boldsymbol{B}$ must be an orthogonal matrix: $\boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}$. If we denote the orthogonal group as

$$SO(m) = \{\boldsymbol{M} \in \mathbb{R}^{m\times m} | \boldsymbol{M}\boldsymbol{M}^T = \boldsymbol{M}^T\boldsymbol{M} = \boldsymbol{I}\}, \tag{8}$$

then the matrix $\boldsymbol{B}$ is restricted in $SO(m)$. By performing the whitening process, we can estimate the matrix $\boldsymbol{B}$ instead of $\boldsymbol{W}$, and the orthogonal constraint of $\boldsymbol{B}$ makes the estimation more accurate. In particular, to estimate the matrix $\boldsymbol{B}$, we consider the log-likelihood function:

$$\begin{aligned} L(\boldsymbol{B}) &= \sum_{j=1}^{n} log f_{\boldsymbol{z}}(\boldsymbol{z}_j) \\ &= \sum_{j=1}^{n}\left[\sum_{i=1}^{m} log f_i(\boldsymbol{b}_i^T\boldsymbol{z}_j) + log\, det(\boldsymbol{B})\right] \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} log f_i(\boldsymbol{b}_i^T\boldsymbol{z}_j), \end{aligned} \tag{9}$$

where $\boldsymbol{b}_i$ is the ith column of $\boldsymbol{B}$, $det(\boldsymbol{B}) = 1$ since $\boldsymbol{B}$ is the orthogonal matrix, $f_{\boldsymbol{z}}$ is the PDF of $\boldsymbol{z}_j$, $j = 1,2,\cdots,n$, and f_i, $i = 1,2,\dots,m$ is the PDF of the ith component that should be extracted [13]. Based on Theorem 1, k out of m density functions should be the density function of the true k independent components, while the rest should be the density function of the standard Gaussian random variable.

Because the density functions f_i, $i = 1,2,\dots,m$, in (9) are unknown, various methods have been proposed to estimate them [13][33][34]. In this study, we choose the kernel density estimation proposed by [13] because it is nonparametric, flexible, and easy to understand. Another advantage of this method is that we can integrate the optimization problem and the estimation of the density function, which are usually separated in other approaches. Specifically, the density functions can be estimated as

$$\hat{f}_i(u_i) = \frac{1}{nh}\sum_{j=1}^{n} \phi\left(\frac{u_i - \boldsymbol{b}_i^T\boldsymbol{z}_j}{h}\right), i = 1,2,\dots,m, \tag{10}$$

where h is the bandwidth of the kernel and ϕ is the Gaussian kernel:

$$\phi(u) = \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{u^2}{2}\right). \tag{11}$$

With the results above, the optimization can be formulated as

$$\begin{aligned} &(P1)\ Max_{\boldsymbol{B}}\ L(\boldsymbol{B}) \\ &= \sum_{i=1}^{m}\sum_{j_1=1}^{n} \log\left[\frac{1}{nh}\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T\boldsymbol{z}_{j_1} - \boldsymbol{b}_i^T\boldsymbol{z}_{j_2}}{h}\right)\right] \end{aligned} \tag{12}$$

$$\text{s.t. } \boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}. \tag{13}$$

In this study, we choose $h = 1.06n^{-1/5}$, as suggested by [35] that can achieve the optimal convergence rate. The optimization problem (P1) is challenging since the feasible region for $\boldsymbol{B}$ is a smooth manifold $SO(m)$. To address this issue, we follow [16] and adopt a Newton-like optimization method on the manifold, as this algorithm has been shown to have a faster (i.e., quadric) convergence rate. Specifically, for the current solution $\boldsymbol{B}_l$ in the l th iteration, there exists a local smooth mapping $\mu_{\boldsymbol{B}_l}: \mathbb{R}^{m(m-1)/2} \to SO(m)$ such that $\mu_{\boldsymbol{B}_l}(\boldsymbol{0}) = \boldsymbol{B}_l$. It follows that the function composition $L \circ \mu_{\boldsymbol{B}_l}$ maps Euclidean space $\mathbb{R}^{m(m-1)/2}$ to $\mathbb{R}$. Then, to update the solution $\boldsymbol{B}_l$, we can apply the Newton method to $L \circ \mu_{\boldsymbol{B}_l}$ and project the result back to the manifold $SO(m)$. The illustration of one iteration of this method is shown in Fig. 1. Thus, the key difference between the Newton-like method and Newton's method is that in the Newton-like method, forward mapping and backward mapping are added before and after the update step of Newton's method in each iteration. Note that this Newton-like method only converges to a local optimal solution as the objective function is not concave. To mitigate this issue, we initialize the solution based on other ICA algorithms, such as *FastICA*, which can also speed up the convergence and reduce the computation burden. The detailed implementation of this method is described in Appendix B.

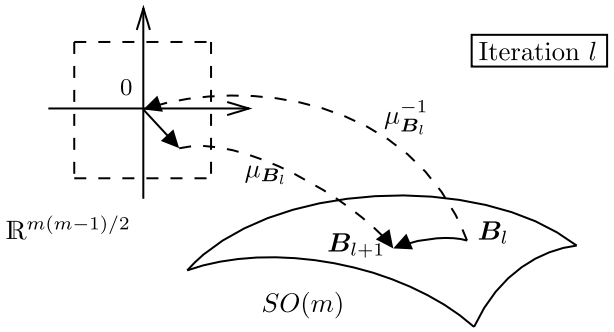


Fig. 1. Illustration of one iteration of the Newton-like method on $SO(m)$ (modified from Fig. 1 in [16]).

B. Transfer Learning-based Independent Component Analysis

In this section, we introduce the proposed MICA algorithm for independent component estimation in the target domain.

As we mentioned before, the potential useful knowledge is the shared component distributions of the source and target domains. If an oracle of the shared distributions is available, we can simply transfer the knowledge by replacing the kernel density estimations in the objective function (12) with the corresponding true densities. However, this oracle is unknown

in practice, and since the density estimation by the ICA may not be accurate, it is challenging to find out all the matched components at one time. To address this issue, we propose to extract only one component in the target domain in each iteration. Specifically, we first extract the component by the conventional ICA (i.e., ICA without transfer learning). Then, we try to find the matched component from the source domain and update the estimation of the conventional ICA in the target domain if such matched component exists.

In particular, there are four major steps in the proposed method: 1) estimate the cumulative density function (CDF) and PDF of the non-Gaussian components in the source domain; 2) extract the independent component in the target domain with the largest negentropy; 3) decide whether a matched component exists in the source domain; and 4) update the estimation of the current independent component in the target domain by transferring the distribution information of the matched component. Next, we will introduce each step in detail.

B.1 Calculate the CDF and PDF of the non-Gaussian components in the source domain

To transfer knowledge, the first step is to solve the ICA problem in the source domain using the method in Section IV.A. Then the PDF and CDF of all independent components in the source domain can be estimated. For the ith component in the source domain, let $\hat{f}_i^{(s)}$ denote the estimated PDF by the kernel density method mentioned in section IV.A, and let $\hat{F}_i^{(s)}$ denote the empirical CDF. Since only non-Gaussian density functions are useful for ICA, the deviation from the Gaussian distribution of each independent component should be analyzed. One of the most widely used measures of non-Gaussianity is negentropy, which is defined for a random variable x as

$$J(f_x) = H(\varphi_x) - H(f_x) = ln(\sigma\sqrt{2\pi e}) - H(f_x), \quad (14)$$

where f_x is the PDF of x, φ_x is the PDF of a Gaussian distribution that has the same variance as x, and $H(f_x)$ is the differential entropy of f_x:

$$H(f_x) = -\int_{-\infty}^{\infty} f_x(u)\log f_x(u)du\,. \quad (15)$$

A well-known theorem stated by Shannon is that when the variance is fixed, the Gaussian random variable has the largest entropy amongst all random variables, which indicates the negentropy of a random variable is always positive. In addition, the smaller the value is, the more similar this random variable is to the Gaussian random variable.

Because the true density function of the source domain components is unknown, we estimate the negentropy from the data. Let $\hat{\boldsymbol{s}}_i^{(s)} = \left[\hat{s}_{i1}^{(s)}, \hat{s}_{i2}^{(s)}, \cdots, \hat{s}_{in_s}^{(s)}\right]$ denote the collection of estimated samples of the ith component. The negentropy of this component can be calculated as

$$\hat{J}\left(\hat{f}_i^{(s)}\right) = ln\left(\sqrt{2\pi e}\right) - \hat{H}\left(\hat{f}_i^{(s)}\right)$$
$$= ln\left(\sqrt{2\pi e}\right) - \frac{1}{n_s}\sum_{j=1}^{n_s}\log \hat{f}_i^{(s)}\left(\hat{s}_{ij}^{(s)}\right). \quad (16)$$

According to our preliminary experiments, the estimation (16) is reliable when the sample size is no less than 150. This estimation can then be used as the test statistic to decide whether the null hypothesis that this component follows the Gaussian distribution. If the estimated negentropy is larger than a predefined threshold, then we reject the null hypothesis and conclude the component has a non-Gaussian distribution. Due to the difficulties in deciding the distribution of the test statistic under the null hypothesis, a Monte-Carlo simulation is conducted to decide the threshold for a given significance level. Specifically, in each round, we sample n_s standard Gaussian random variables and estimate the negentropy using (16). Then this procedure is repeated N times. In practice, $N = 200$ should be large enough. Then we have obtained N samples of the negentropy under the null hypothesis. For a given significance level α, the threshold of the hypothesis test can be estimated as the $1-\alpha$ quantile of the N negentropy values. It is worth mentioning that this Monte-Carlo simulation can be conducted offline and will not increase the computation load of the MICA. One can also choose other methods to test the Gaussianity, such as Shapiro-Wilk test and Carmer-von Mises test. All these tests are powerful as long as there are enough data in the source domain. After this test, only the CDFs and PDFs of the non-Gaussian components are kept. We denote the collection of non-Gaussian CDFs as set P_{CDF}.

B.2 Extract the component with the largest negentropy

In the next step, we first extract one component from the target domain using ICA without transfer learning. A question comes up immediately: which component to extract in each round?

To answer this question, we propose to extract the component with the largest non-Gaussianity, where the non-Gaussianity is measured by the estimated negentropy. There are two main reasons. First, it is well known that the component with larger non-Gaussianity is easier to be separated in ICA. In an extreme case, when all components follow the Gaussian distribution, it will be impossible to estimate them as discussed in Section III. In other words, the components estimated with larger negentropy are generally more accurate in ICA [9]. Second, it is proved in [37] that the linear combination of independent variables will lead to smaller negentropy. Specifically, let v_1 and v_2 denote two independent random variables with the same variance σ^2. Then $v_3 = a_1v_1 + a_2v_2$ also has the same variance σ^2 if $a_1^2 + a_2^2 = 1$, and it is proved that $J(a_3) \le max\{J(a_1), J(a_2)\}$. Therefore, the estimated components with smaller negentropy are more likely to be a linear combination of some true components, which again indicates the estimated component with larger negentropy should be more accurate.

Now let us discuss how we can obtain the estimation of the components with the largest negentropy in the target domain. Let $\widehat{\boldsymbol{S}}_0^{(t)} \in \mathbb{R}^{m_t \times n_t}$ denote the matrix that contains the initial estimation of the m_t independent components in the target domain by applying the ICA in Section IV.A, where the unmixing matrix is $\boldsymbol{B}_0$. For each extracted component, its negentropy can be estimated by (16). Then we can permute the

rows of the matrix $\widehat{\boldsymbol{S}}_0^{(t)}$ such that the component in the first row has the largest negentropy value. Note that the rows of the unmixing matrix $\boldsymbol{B}_0$ should also be permuted accordingly. After the permutation, the first row of $\widehat{\boldsymbol{S}}_0^{(t)}$ represents the estimated component with the largest negentropy.

B.3 Decide whether a matched component exists

In this step, we will decide if a component from the source domain is matched with the component we extracted in the previous step, and if so, we further transfer the knowledge. Let $\hat{s}_0^{(t)}$ denote the estimated component that has the largest negentropy and $\hat{F}_0^{(t)}$ denote the corresponding empirical CDF, where the subscript 0 refers to the component extracted in the first iteration. To analyze the distance of $\hat{F}_0^{(t)}$ to the CDFs in the set P_{CDF}, the supremum distance can be used. In particular, for $\hat{F}_j^{(s)} \in P_{CDF}$, the distance can be calculated as

$$d_{0,j} = sup_u \left|\hat{F}_0^{(t)}(u) - \hat{F}_j^{(s)}(u)\right|. \tag{17}$$

Then, the most similar component in the source domain is

$$j^* = arg\, min_j\, d_{0,j}. \tag{18}$$

To decide whether the component $s_{j^*}^{(s)}$ and $\hat{s}_0^{(t)}$ share the same distribution, the two-sample Kolmogorov–Smirnov test can be conducted. Specifically, the supremum distance d_{0,j^*} is the test statistic and for a given significance level α, the threshold can be calculated as $\delta = \sqrt{log\left(\frac{2}{\alpha}\right)\frac{n_s+n_t}{2n_s n_t}}$. If $d_{0,j^*} > \delta$, we conclude that $s_{j^*}^{(s)}$ and $\hat{s}_0^{(t)}$ have different distributions. Otherwise, components $s_{j^*}^{(s)}$ and $\hat{s}_0^{(t)}$ are considered matched and share the same distribution. When domain knowledge is available, α can be chosen accordingly. For example, if we have evidence that lots of components are shared by the two domains, then smaller α should be used to transfer more knowledge. Otherwise, large α can be used to prevent negative transfer. If such domain knowledge is not available, we suggest using $\alpha = 0.2$ according to the numerical experiments.

B.4 Update the estimation of the independent component by transferring knowledge

There are two possible outcomes of the hypothesis in the previous step. We will discuss the two scenarios individually.

On one hand, when the matched component in the source domain can be found, we leverage the density function of that component by replacing the density function f_i in the likelihood function (12) with the density function $\hat{f}_{j^*}^{(s)}$ of the matched component. The optimization problem is then formulated as

$$(P2)\ Max_{\boldsymbol{B}}\ L(\boldsymbol{B}) = \sum_{i=2}^{m_t}\sum_{j_1=1}^{n_t} \log\left[\frac{1}{n_t h}\sum_{j_2=1}^{n_t}\phi\left(\frac{\boldsymbol{b}_i^T\boldsymbol{z}_{j_1} - \boldsymbol{b}_i^T\boldsymbol{z}_{j_2}}{h}\right)\right] + \frac{1}{n_t}\sum_{j_1=1}^{n_t}\log \hat{f}_{j^*}^{(s)}\left(\boldsymbol{b}_1^T\boldsymbol{z}_{j_1}\right), \tag{19}$$

$$\text{s.t. } \boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}. \tag{20}$$

A similar geometric Newton-like method as introduced in Appendix B can be applied to solve this problem, where $\boldsymbol{B}$ can be initialized as $\boldsymbol{B}_0$ to accelerate the convergence. Let $\boldsymbol{B}_{0,new}$ and $\widehat{\boldsymbol{S}}_{0,new}^{(t)}$ denote the updated unmixing matrix and the estimated components. Since the density function $\hat{f}_{j^*}^{(s)}$ has been used, we will remove the corresponding CDF from the set P_{CDF}.

On the other hand, if we cannot find a matched component, then we will denote $\boldsymbol{B}_{0,new} = \boldsymbol{B}_0$ and $\widehat{\boldsymbol{S}}_{0,new}^{(t)} = \widehat{\boldsymbol{S}}_0^{(t)}$, as this is already the best estimation that we can get for the first component. With the solution above, we can extract the first component as the first row of $\widehat{\boldsymbol{S}}_{0,new}^{(t)}$. We let $\widehat{\boldsymbol{S}}_{0,new}^{(t)} = \begin{bmatrix}\widehat{\boldsymbol{p}}_0^{(t)} \\ \widehat{\boldsymbol{S}}_1^{(t)}\end{bmatrix}$, where $\widehat{\boldsymbol{p}}_0^{(t)} \in \mathbb{R}^{1\times n_t}$ denotes the first row of the matrix $\widehat{\boldsymbol{S}}_{0,new}^{(t)}$ and $\widehat{\boldsymbol{S}}_1^{(t)} \in \mathbb{R}^{(m_t-1)\times n_t}$ denotes the rest of the matrix $\widehat{\boldsymbol{S}}_{0,new}^{(t)}$. In this way, we have finished the extraction of the first component in the target domain.

B.5 Summary of the MICA

Previously we have introduced the four major steps in the proposed MICA algorithm. For step 2 to step 4, we describe the implementation details for the first iteration (i.e., extraction of the first component from the target domain). To extract all the remaining independent components in the target domain, we need to repeat step 2 to step 4 multiple times. Namely, for the second iteration, $\widehat{\boldsymbol{S}}_1^{(t)}$ can be used as the whitened data matrix and another independent component can be extracted in the same manner. We will repeat this procedure until m_t components are extracted. This algorithm can also stop earlier when all the remaining components are found to be Gaussian. Let $\boldsymbol{B}_{l,new} \in \mathbb{R}^{(m_t-l)\times(m_t-l)}$ denote the unmixing matrix estimated in iteration l, $\widetilde{\boldsymbol{B}}_{l,new} = \begin{bmatrix}\boldsymbol{I}_l & \boldsymbol{0} \\ \boldsymbol{0}^T & \boldsymbol{B}_{l,new}\end{bmatrix} \in \mathbb{R}^{m_t\times m_t}$ if $l > 0$ and $\widetilde{\boldsymbol{B}}_{0,new} = \boldsymbol{B}_{0,new}$. Then for the first two iterations, we have,

$$\begin{bmatrix}\widehat{\boldsymbol{p}}_0^{(t)} \\ \widehat{\boldsymbol{S}}_1^{(t)}\end{bmatrix} = \widehat{\boldsymbol{S}}_{0,new}^{(t)} = \left(\widetilde{\boldsymbol{B}}_{0,new}\right)^T\widehat{\boldsymbol{S}}_0^{(t)}, \tag{21}$$

$$\begin{bmatrix}\widehat{\boldsymbol{p}}_0^{(t)} \\ \widehat{\boldsymbol{p}}_1^{(t)} \\ \widehat{\boldsymbol{S}}_2^{(t)}\end{bmatrix} = \begin{bmatrix}\widehat{\boldsymbol{p}}_0^{(t)} \\ \widehat{\boldsymbol{S}}_{1,new}^{(t)}\end{bmatrix} = \begin{bmatrix}1\times\widehat{\boldsymbol{p}}_0^{(t)} \\ \left(\boldsymbol{B}_{1,new}\right)^T\widehat{\boldsymbol{S}}_1^{(t)}\end{bmatrix} = \left(\widetilde{\boldsymbol{B}}_{1,new}\right)^T\widehat{\boldsymbol{S}}_{0,new}^{(t)} = \left(\widetilde{\boldsymbol{B}}_{0,new}\widetilde{\boldsymbol{B}}_{1,new}\right)^T\widehat{\boldsymbol{S}}_0^{(t)}. \tag{22}$$

Thus, we can keep this process until all the non-Gaussian independent components are extracted. If there are k_t non-Gaussian components in the target domain, the final mixing matrix $\boldsymbol{B}$ can then be derived as

$$\boldsymbol{B} = \widetilde{\boldsymbol{B}}_{0,new}\widetilde{\boldsymbol{B}}_{1,new}\cdots\widetilde{\boldsymbol{B}}_{k_t-1,new}. \tag{23}$$

The graphical diagram of the MICA algorithm is shown in Fig. 2.

V. Simulation

In this section, we conduct simulations based on four different settings to fully evaluate the performance of the proposed MICA method. To the best of our knowledge, there is

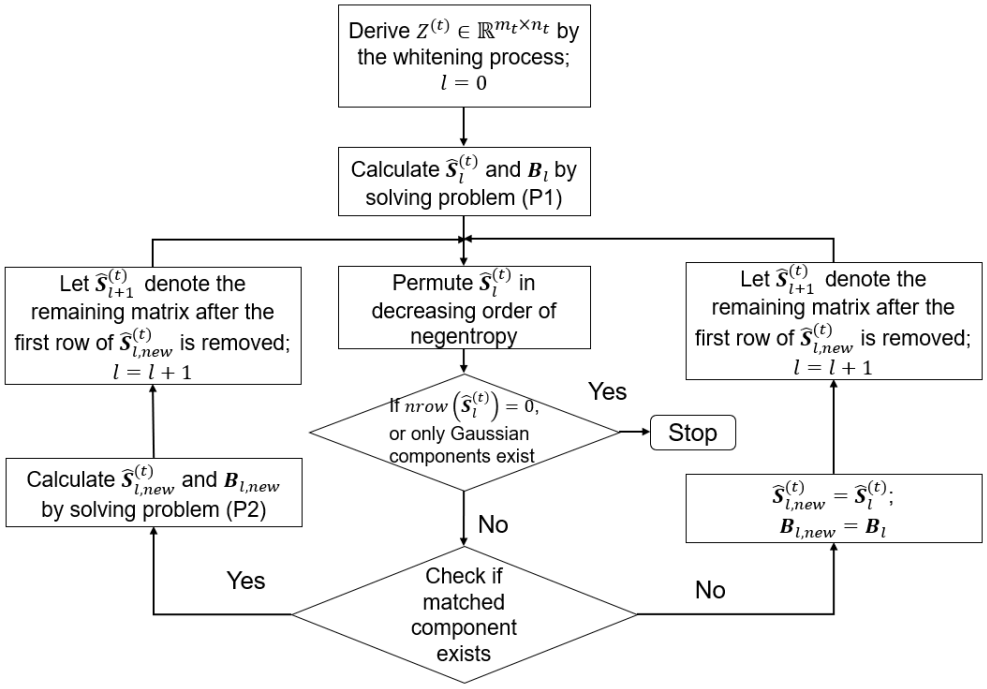


Fig. 2. Flowchart of the proposed method.

no existing literature that can deal with different mixing matrices in the source and the target domains. Thus, we demonstrate the benefit of using MICA by comparing its performance to the conventional ICA. Here, the conventional ICA refers to the ICA without transfer learning, when only target domain data are used. The model settings and the results are discussed below.

A. Setting I

In this subsection, we investigate the performance of MICA with different n_t and noise levels when the number of independent components is 2 in both the source and the target domains. In particular, it is assumed that $\boldsymbol{X}^{(s)}, \boldsymbol{S}^{(s)} \in \mathbb{R}^{2\times n_s}$, $\boldsymbol{X}^{(t)}, \boldsymbol{S}^{(t)} \in \mathbb{R}^{2\times n_t}$ and $\boldsymbol{e}_j^{(d)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ for $d \in \{s,t\}$, $j = 1,2,\dots,n_d$. Besides, it is also assumed that $\boldsymbol{s}_{1j}^{(d)} \overset{\text{i.i.d}}{\sim} Unif(-\sqrt{3},\sqrt{3})$, and $\boldsymbol{s}_{2j}^{(d)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$, for $d \in \{s,t\}$, $j = 1,2,\dots,n_d$, which means the distributions of the two independent components in the two domains are the same. Since there is one component in both domains sharing the same uniform distribution, the component's density function from the source domain should be transferred to improve the estimation accuracy of the ICA in the target domain. Note that the Gaussian density should not be transferred.

To evaluate the accuracy of the estimated mixing matrix $\widehat{\boldsymbol{A}}^{(t)}$ for the target domain, we need to compare it with the true mixing matrix $\boldsymbol{A}^{(t)}$. As mentioned before, the mixing matrix $\boldsymbol{A}^{(t)}$ can only be estimated up to a permutation of columns. Therefore, we use the Amari metric to evaluate the accuracy of $\widehat{\boldsymbol{A}}^{(t)}$, which is given as

$$d\left(\widehat{\boldsymbol{A}}^{(t)}, \boldsymbol{A}^{(t)}\right) = \frac{1}{2m}\sum_{i=1}^{m}\frac{\sum_{j=1}^{m}|r_{ij}|}{max_j\, r_{ij}} + \frac{1}{2m}\sum_{j=1}^{m}\frac{\sum_{i=1}^{m}|r_{ij}|}{max_i\, r_{ij}} - 1, \quad (24)$$

where r_{ij} is the ijth entry of matrix $\boldsymbol{R} = \left(\widehat{\boldsymbol{A}}^{(t)}\right)^{-1}\boldsymbol{A}^{(t)}$, m is the number of rows of $\boldsymbol{A}^{(t)}$ and equals 2 in this setting. This metric is appropriate to be used here as it is invariant to the permutation of the columns of $\widehat{\boldsymbol{A}}^{(t)}$ and $\boldsymbol{A}^{(t)}$. When $d\left(\widehat{\boldsymbol{A}}^{(t)}, \boldsymbol{A}^{(t)}\right)$ is small, it means that the extracted components are close to the true components. Otherwise, if $d\left(\widehat{\boldsymbol{A}}^{(t)}, \boldsymbol{A}^{(t)}\right)$ is large, the extracted components have large errors.

In the first experiment, we consider the influence of n_t. In this simulation, we fix $n_s = 2000$, $\sigma = 0$, and let n_t take values from 50 to 350. For each n_t value, 500 simulation runs are conducted and in each single simulation, the entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are independently drawn from the distribution $Unif[-1,1]$. Then the proposed MICA and conventional ICA are applied to the data and the estimation of the mixing matrix in the target domain can be obtained. The average of the Amari errors of these two methods in 500 simulation runs is recorded. To get a better understanding of the improvement of the proposed MICA algorithm, we consider the relative error decrease ratio of each n_t value. Let $\bar{e}_{\text{ICA}}(n_t)$ and $\bar{e}_{\text{MICA}}(n_t)$ denote the average Amari errors of the two methods with value n_t. Then the relative error decrease ratio of MICA is defined as

$$r(n_t) = \frac{\bar{e}_{\text{ICA}}(n_t) - \bar{e}_{\text{MICA}}(n_t)}{\bar{e}_{\text{ICA}}(n_t)}. \quad (25)$$

The average Amari error of the two methods and the relative error decrease ratio with different n_t values are shown in Fig. 3.

From Fig. 3 we can see that when n_t increases, the Amari errors of both MICA and ICA decrease, which is consistent with our expectations. Besides, the MICA is uniformly better than the conventional ICA. This implies that by transferring the density function from the source domain, the estimation accuracy in the target domain can be improved. Comparing the

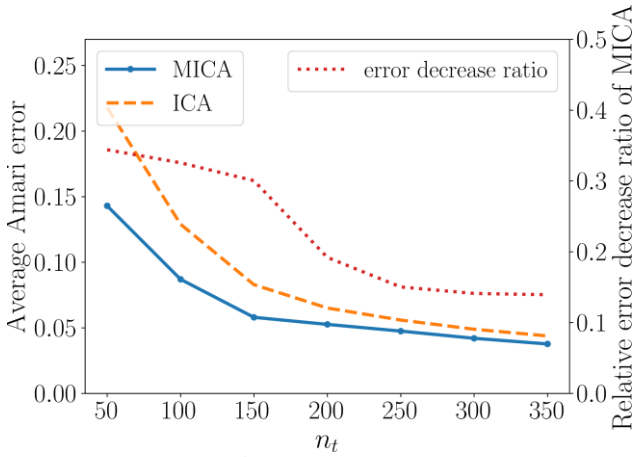


Fig. 3. Average Amari error of MICA and ICA and relative error decrease ratio of MICA with different n_t.

relative error decrease ratio of MICA with different n_t, we can observe that when n_t is small, the improvement is more significant. For example, when $n_t \leq 150$, the average Amari error of MICA is 30% less than that of ICA. This is because when the number of training samples in the target domain is small, we cannot accurately estimate the density functions of the independent components by the conventional ICA. Thus, the benefit of transferring density functions from the source domain will be significant. On the contrary, when n_t is large, the density functions estimated in the target domain are getting close to the true densities, and thus it becomes less beneficial to transfer density functions from the source domain to the target domain. Nevertheless, even when $n_t \geq 250$, the Amari error still decreases around 15% when MICA is applied, which again shows the superiority of the proposed method.

In the next experiment, we fix n_t to be 100 and consider the Amari error when σ varies. Like before, $n_s = 2000$ and we consider the average Amari error in 500 simulation runs, where the entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are independently drawn from the distribution $Unif[-1,1]$ in each repeated simulation. The results are summarized in Fig. 4.

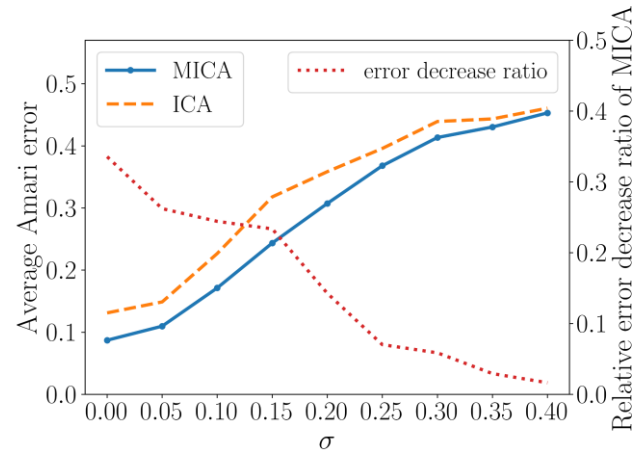


Fig. 4. Average Amari error of MICA and ICA and relative error decrease ratio of MICA with different σ.

As expected, the Amari errors of both methods increase as σ increases. In this experiment, we can also see that the MICA outperforms the conventional ICA for every σ value, especially when σ is small. One possible reason is that when the standard deviation of the noise term is larger in the source domain, the estimated density functions are less accurate. Thus, if these density functions are transferred, the improvement of mixing matrix estimation is limited. Another possible reason is that if σ in the target domain is very large, our initial estimation derived by ICA may have a large error. In such a case, our algorithm may lead to a wrong conclusion that there is no independent component in the source domain that has the same distribution as the components extracted in the target domain and thus no knowledge will be transferred. In our experiments, indeed when $\sigma = 0.4$, the knowledge is only transferred in 23.31% of the simulation runs. Despite the limitation when σ is large, the improvement is significant when applying MICA with $\sigma \leq 0.15$, where the relative error decrease ratio of MICA is larger than 20%.

B. Setting II

In the first setting, we only consider the case that the two independent components follow the uniform and Gaussian distributions. In this setting, more distributions are explored. Specifically, it is assumed that $\boldsymbol{X}^{(s)}, \boldsymbol{S}^{(s)} \in \mathbb{R}^{2\times n_s}$, $\boldsymbol{X}^{(t)}, \boldsymbol{S}^{(t)} \in \mathbb{R}^{2\times n_t}$ and $\boldsymbol{e}_{\cdot j}^{(d)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(\boldsymbol{0}, 0.1^2\boldsymbol{I})$ for $d \in \{s,t\}$, $j = 1,2,\dots,n_d$. Besides, n_s and n_t are fixed to be 2000 and 200, respectively. We also assume that the two independent component distributions in the source domain and the target domain are the same, where the two distributions are selected from the following five distributions:

(1) Exponential: $exp(1)$.
(2) Multimodal: $f = \frac{1}{2}\mathcal{N}(0.9,1) + \frac{1}{2}\mathcal{N}(-0.9,1)$, where $\mathcal{N}$ is the density function of the Gaussian distribution.
(3) Uniform: $Unif(-\sqrt{3},\sqrt{3})$.
(4) Gamma: $Gamma(3,2)$, where 3 is the shape parameter and 2 is the scale parameter.
(5) Gaussian: $\mathcal{N}(0,1)$.

The negentropy measures of these five distributions estimated by 5000 random samples are listed in TABLE 1.

TABLE 1
NEGENTROPY OF THE FIVE DISTRIBUTIONS

Distribution	Negentropy
Exponential	0.3288
Multimodal	0.2057
Uniform	0.1244
Gamma	0.1167
Gaussian	0.0021

In the experiment, we consider each possible pair of independent component distributions except both independent components follow the Gaussian distribution. This is because the two components are not identifiable when their distributions are both Gaussian, as mentioned in Section III. However, the proposed MICA method and conventional ICA can extract all the components when there is only one Gaussian independent component, since after all the non-Gaussian independent components are extracted, we can directly obtain the last component, which is the Gaussian component. Next, for each

possible pair of distributions, the average Amari errors of ICA and MICA in 500 simulation runs are shown in TABLE 2. Again, the entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are independently drawn from the distribution $Unif[-1,1]$ in each simulation run.

TABLE 2
AVERAGE AMARI ERRORS OF MICA AND ICA WITH DIFFERENT INDEPENDENT COMPONENT DISTRIBUTIONS

Component Distributions		ICA	MICA	Relative error decrease ratio
Exponential	Exponential	0.125	0.112	9.83%
Exponential	Multimodal	0.111	0.105	5.40%
Exponential	Uniform	0.110	0.106	4.14%
Exponential	Gamma	0.130	0.114	12.35%
Exponential	Gaussian	0.150	0.142	5.59%
Multimodal	Multimodal	0.101	0.095	6.10%
Multimodal	Uniform	0.117	0.106	9.57%
Multimodal	Gamma	0.119	0.110	7.45%
Multimodal	Gaussian	0.148	0.134	9.46%
Uniform	Uniform	0.128	0.110	14.06%
Uniform	Gamma	0.124	0.111	10.46%
Uniform	Gaussian	0.218	0.162	25.69%
Gamma	Gamma	0.188	0.139	26.32%
Gamma	Gaussian	0.234	0.182	22.25%

From TABLE 2, we can see that for all the distribution pairs, the average Amari error of MICA is less than that of ICA, showing that the proposed method can handle different kinds of component distributions. The main reason is that we choose to use the nonparametric kernel density estimation, which is very flexible to fit the density. Another interesting finding is that when both component distributions are selected from those distributions with small negentropy measures, e.g., uniform distribution, gamma distribution, and Gaussian distribution, the error decrease ratio by using transfer learning is generally more significant. The reason is that when the distributions of both independent components are closer to the normal distribution (i.e., smaller negentropy measures), it is more difficult for the conventional ICA to separate them based on the data in the target domain and the density function estimations may have larger errors. Therefore, the transferred density function is more beneficial in this case, as shown in the last five rows of TABLE 2. Likewise, if both distributions have large negentropy, the estimated unmixing matrices and density functions are more accurate. And thus, it becomes less beneficial to transfer densities in this case.

C. Setting III

In this setting, we further test the proposed MICA when we have more independent components in both the source and the target domains. It is assumed that for $d \in \{s,t\}$ and $j = 1,2,\dots,n_d$, $\boldsymbol{X}^{(s)}, \boldsymbol{S}^{(s)} \in \mathbb{R}^{5\times 2000}$, $\boldsymbol{X}^{(t)}, \boldsymbol{S}^{(t)} \in \mathbb{R}^{5\times n_t}$ and $\boldsymbol{e}_{\cdot j}^{(d)} \overset{\text{i.i.d}}{\sim} \mathcal{N}(\boldsymbol{0}, 0.05^2\boldsymbol{I})$. For the target domain, we also assume that the five independent components follow the five distributions described in setting II. To demonstrate the advantages of the proposed method, we consider the following three scenarios:

(1) Scenario 1: Only the target domain data are used, and the conventional ICA is applied.
(2) Scenario 2: The first three independent components in the source domain follow the standard Gaussian distribution. The other two independent components follow the Beta distribution with parameters (0.5,0.5). Then we apply the MICA to transfer knowledge from the source domain to the target domain. Note that in this case, no matched components should be found in the source and the target domains, and thus no knowledge should be transferred.
(3) Scenario 3: The five independent components in the source domain also follow the same five distributions in setting II. Then, MICA is applied to the data.

Note that the target domain data remains the same in all three scenarios above. In this numerical experiment, we consider two possible n_t values: 100 and 200. Then for each value of n_t, the simulation is repeated 200 times, while in each simulation, the entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are generated in the same manner as before. The boxplot of the Amari error is shown in Fig. 5. When MICA is applied in scenario 2 and scenario 3, we summarize the identified number of matched components during the 200 simulation runs in TABLE 3.

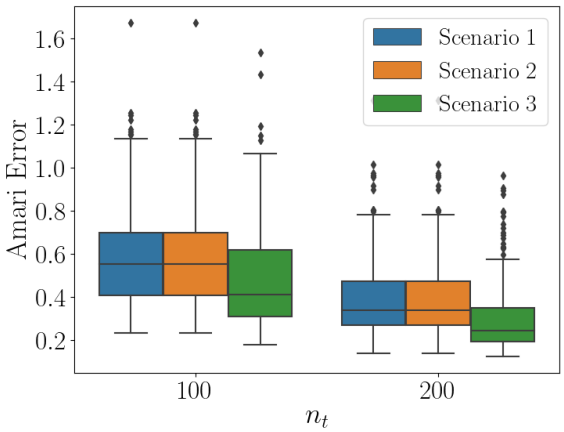


Fig. 5. Boxplot of the Amari error in the three scenarios when n_t is 100 and 200.

TABLE 3
NUMBER OF MATCHED COMPONENTS IDENTIFIED IN SCENARIOS 2 AND 3 DURING THE 200 SIMULATION RUNS

Scenario	n_t	Number of transferred components					
		0	1	2	3	4	5
2	100	200	0	0	0	0	0
	200	200	0	0	0	0	0
3	100	7	4	16	58	115	0
	200	10	8	21	68	93	0

We first compare scenario 1 and scenario 3. From the boxplot, we can observe that for both n_t values, the performance of MICA (scenario 3) is much better than ICA (scenario 1) when some density functions can be transferred from the source domain to the target domain. Since there are four matched non-Gaussian components, ideally four components in the source domain should be identified as matched components and transferred. Based on the results of scenario 3 in TABLE 3, at least one matched component from the source domain is found and transferred in almost all simulation runs. Besides, for more than 80% of the simulation runs, at least three non-gaussian matched components from the source domain are correctly identified. These results justify the

effectiveness of the proposed MICA method.

Next, let us compare the results of scenario 1 and scenario 2. Recall that in scenario 2, the only common distribution shared in the source and target domains is the normal distribution. Therefore, no density function should be transferred, and any knowledge transfer belongs to the negative transfer, which will increase the estimation error in the target domain. From Fig. 5 and TABLE 3, we can see that indeed no knowledge is transferred and MICA generates the same results as ICA in all simulation runs. Therefore, the negative transfer is avoided by using MICA. There are two reasons that there is no negative transfer in scenario 2: (i) we only transfer non-Gaussian distributions from the source domain according to section IV.B.1. (ii) More importantly, when MICA is applied, knowledge transfer only occurs when matched components from the source domain can be found. This result demonstrates that the proposed method helps reduce negative transfer when no useful knowledge is available.

D. Setting IV

In this setting, we further evaluate the proposed method when $m_s \neq m_t$, $k_s \neq k_t$, and entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are generated from different distributions. To make the content easy to understand, we adopt all the assumptions in setting III. The only differences are that: (i) for the target domain, it is assumed that $\boldsymbol{X}^{(t)}, \boldsymbol{S}^{(t)} \in \mathbb{R}^{7\times n_t}$. Among the seven independent components, five of them are the same as before and follow the five distributions described in setting II. In addition, there are two extra exponentially distributed components. (ii) The entries of $\boldsymbol{A}^{(t)}$ are independently sampled from an exponential distribution with a mean of 1. The Amari error of the three scenarios with different n_t values, as well as the number of the matched components identified in scenarios 2 and 3 during the 200 simulation runs, are shown in Fig. 6 and TABLE 4. The results are similar to the results in Fig. 5, which further validates the proposed method when $m_s \neq m_t$, $k_s \neq k_t$, and entries of $\boldsymbol{A}^{(s)}$ and $\boldsymbol{A}^{(t)}$ are generated from different distributions.

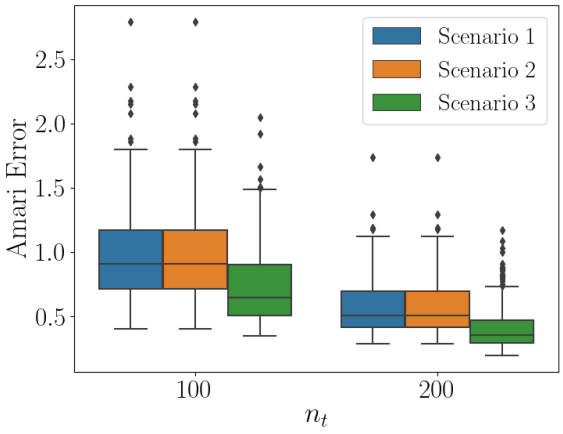


Fig. 6. Boxplot of the Amari error in the three scenarios when n_t is 100 and 200.

TABLE 4
NUMBER OF MATCHED COMPONENTS IDENTIFIED IN SCENARIOS 2 AND 3 DURING THE 200 SIMULATION RUNS

Scenario	n_t	Number of transferred components					
		0	1	2	3	4	5
2	100	200	0	0	0	0	0
	200	200	0	0	0	0	0
3	100	2	6	7	35	150	0
	200	10	15	18	33	124	0

VI. CASE STUDY

In this section, we implement the proposed method to extract fetal ECG signals from maternal ECG records. Every year, one out of 125 babies were born with some heart defects [38]. Therefore, analyzing the fetal ECG and detecting the disease before birth is of great importance. However, directly measuring the fetal ECG from the fetal scalp is not desired as it not only poses risk to the fetus, but also can only be conducted after a certain stage [39]. As a result, it is often the practice to obtain the abdominal ECG from a pregnant female, which is much easier and safer. In such a case, ICA has been widely used in fetal ECG extraction based on the collected abdominal ECG signals [3].

In particular, we try to improve fetal ECG extraction by employing the proposed transfer learning method. The results are compared with the benchmark approach: the conventional ICA without transfer learning. When the conventional ICA is applied, only the target domain data are used.

A. Overview of the Data Set

In this case study, we use the ECG data in the Abdominal and Direct Fetal ECG Database of PhysioNet [40], which was first analyzed in [41]. This database contains multi-channel ECG recordings obtained from pregnant women. In this case study, we focus on subjects r01 and r04 of the data set. For each subject, four signals are collected from the maternal abdomen. Besides, one additional channel of ECG signal is directly obtained from the fetal scalp. The sampling rates of each signal are 1kHz. More details of this database can be found in [41].

The original signals of subject r01 in the first ten seconds are shown in Fig. 7. In this figure, channel f is the ECG signal obtained from the fetal scalp and will be used as the ground truth. The rest four channels are collected from the maternal abdomen. Clearly, each signal from the maternal abdomen can

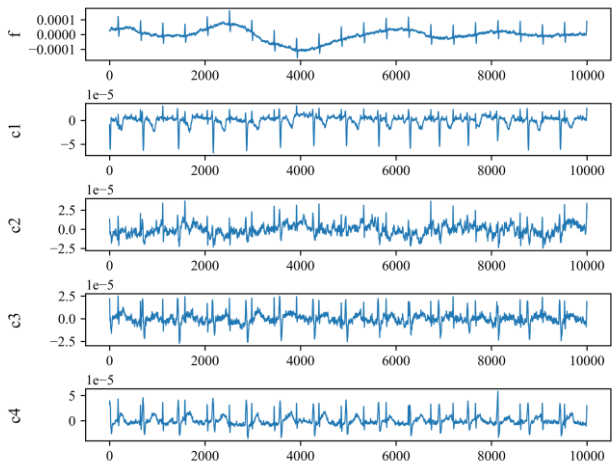


Fig. 7. The original ECG signal of subject r01.

be regarded as a linear combination of maternal ECG signal, fetal ECG signal, and other irrelevant signals. Therefore, the assumptions of ICA are valid in this data set. In addition, due to the subject-to-subject variations, the mixing matrices of different subjects are different. The goal of fetal ECG extraction is to accurately reconstruct the spikes of fetal ECG from these four channels, as the spikes contain important information about fetal health conditions [42].

Before implementing the ICA and the proposed method, the observed signals were preprocessed. Specifically, for each channel, we use the low-pass Butterworth filter to estimate the baseline drift signal. We follow [3] to employ the third-order Butterworth filter and the cut-off frequency is set to be 5Hz. Then the drift signal is removed from each ECG channel.

B. Results and Comparison

In this study, we consider transferring knowledge from subject r01 to subject r04. Specifically, for the source domain r01, it is assumed that the signals obtained in the first ten seconds are available. However, for the target domain r04, to simulate the data sparsity challenge, we downsample the signal in the first ten seconds with a rate of 20. Namely, $n_s = 10000$ and $n_t = 500$. Besides, $m_s = m_t = 4$, as there are four channels in each domain.

To implement our method, we first solve the ICA problem in the source domain. The extracted components and the filtered true fetal ECG of r01 are shown in Fig. 8. From Fig. 8, we can see that the spikes identified in component s1 are well aligned with the ones in the ground truth signal f. Indeed, component s1 refers to the extracted fetal ECG signal and component s4 is the maternal ECG signal, since the normal baseline of fetal heart rate is 110-160 bpm, which is much higher than the maternal heart rate with a baseline of 60-100 BMP [43]. Besides, component s2 contains uneven spikes, which are difficult to explain, and component s3 contains unmeaningful noise. Since enough data are available in the source domain, the fetal ECG extraction is pretty accurate and most of the spikes are corrected and reconstructed in component s1.

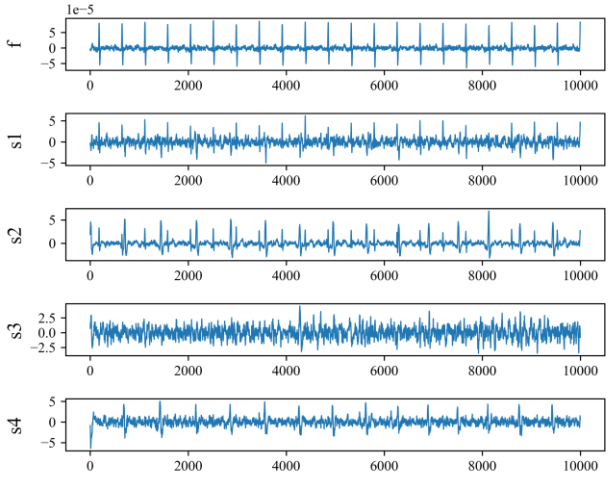


Fig. 8. Extracted components (s1~s4) and filtered fetal ECG (f) of r01.

In the next step, we apply the proposed MICA method to extract fetal ECG in the target domain and compare the result with the conventional ICA. When MICA is applied, the component s1 in Fig. 8 is found to be a matched component. Then the density function of this component is transferred to improve the performance of r04. Since four components are extracted by each method, to compare the performance of MICA and conventional ICA, we select the component that is most similar to the true fetal ECG by each method. The results are shown in Fig. 9.

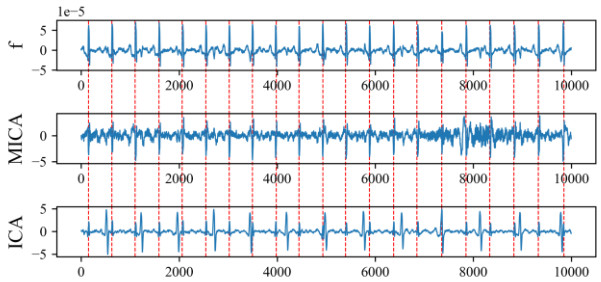


Fig. 9. Extracted fetal ECG signal by MICA and ICA and filtered fetal ECG (f) of r04.

Based on Fig. 9, we can see that by using the conventional ICA, the extracted fetal ECG is largely contaminated by the maternal ECG signal and many spikes in the signal do not align with the spikes in the true fetal ECG signal f. As for the fetal ECG extraction by MICA, most of the ECG spikes are clearly separated from the maternal ECG signal and noise, and they align with the true spikes very well.

VII. Conclusion and Future Work

ICA is widely used to separate multivariate signals into independent components. However, one limitation in the existing ICA methods is that the performance will degrade when the data are limited. To handle this issue, we propose to transfer knowledge from the source domain and improve the component extraction in the target domain. In this approach, the challenges are twofold. One is deciding what knowledge can be transferred. The other is how to reduce the negative transfer when no commonality is shared by the two domains.

To address the challenges above, we proposed a novel transfer learning method tailored to the unique characteristics of ICA. The key idea is to transfer the component distribution information from the source domain to the target domain considering some independent components from the two sources may share the same distribution. Specifically, we first extract all the independent components in the source domain by maximizing the log-likelihood function. Then for the target domain, we extract the component one by one based on the largest negentropy. After we extract one component from the target domain, we try to find a matched component in the source domain and if such matched component can be found, the extraction accuracy is improved by leveraging the PDF of the matched component. Consequently, our method effectively transfers useful knowledge from the source domain to the target domain. Meanwhile, the negative transfer is minimized as no knowledge is transferred when there is no matched component in the two domains. The effectiveness of the proposed method was justified by numerical simulations and a case study with

ECG data.

In future research, several important topics need to be investigated. First, the proposed method only transfers the information of component distributions from the source domain to the target domain. It is desired to explore whether we can transfer some other knowledge and further improve the component extraction accuracy in the target domain. For example, in some cases, the mixing matrices in the source domain and target domain may also share some similarities. Second, it is worth investigating how we can extend the proposed method to nonlinear cases. Under the assumption of ICA, each signal is a linear combination of some latent independent components with some noise. To mitigate the risk of the linear assumption, some methods have been proposed recently to recover the independent components when signals are nonlinear functions of the components [44][45]. Nevertheless, these methods require other rigid assumptions and do not consider the issue of data sparsity. To fill in the literature gap, our current idea is to combine manifold learning and linear ICA. Then the knowledge of component distributions can be transferred, and components can be extracted accurately in nonlinear cases without imposing extra assumptions. Third, in this work, we only focus on one-to-one component matching in both domains. However, it is possible that the matching of components is one-to-many or many-to-many from the source domain to the target domain. Thus, it would be interesting to study how to decide on the matching mode and deal with more complicated scenarios of component matching. Last but not least, though the Newton-like method has low time complexity in general, the computation may be slow when the number of samples or the number of observed variables is extremely large. Therefore, we may consider generalizing the current Newton-like method to a stochastic descent version, which only uses a part of the data to update the solution in each iteration and the computational issue can be largely mitigated.

APPENDIX A
PROOF OF THEOREM 1

Since $\boldsymbol{A} \in \mathbb{R}^{m\times k}$ and $rank(\boldsymbol{A}) = k$, the null space of $\boldsymbol{A}^T$, $Nul(\boldsymbol{A}^T)$, has a dimension $m-k$. We let the column vectors $\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_{m-k}$ be the $m-k$ basis of $Nul(\boldsymbol{A}^T)$. Denote $\boldsymbol{B}_1 = [\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_{m-k}]^T \in \mathbb{R}^{(m-k)\times m}$. Obviously, we have $\boldsymbol{B}_1\boldsymbol{A} = \boldsymbol{0}_{(m-k)\times k}$. Before we continue the proof, two lemmas related to the matrix $\boldsymbol{B}_1$ are stated and proved.

Lemma 1: $rank(\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T) = m-k$, where recalls that $\boldsymbol{U}$ is the covariance matrix of the additive error $\boldsymbol{e}_i$, for $i = 1,2,\cdots,n$.

Proof: We prove this lemma by contradiction. Assume that $rank(\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1) < m-k$, then there must exist $\boldsymbol{c} \in \mathbb{R}^{(m-k)\times 1}$ such that $\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{c} = \boldsymbol{0}$, which indicates that $\boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{c} = \boldsymbol{0}$. Hence, if we let $\boldsymbol{u} = \boldsymbol{B}_1^T\boldsymbol{c}$, for $j = 1,2,\cdots,n$, we have

$$\boldsymbol{u}^T\boldsymbol{x}_j = \boldsymbol{u}^T\boldsymbol{A}\boldsymbol{s}_j + \boldsymbol{u}^T\boldsymbol{e}_j = \boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{A}\boldsymbol{s}_j + \boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{e}_j = \boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{e}_j, \quad (26)$$

where $\boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{A}\boldsymbol{s}_j = \boldsymbol{0}$ because $\boldsymbol{B}_1\boldsymbol{A} = \boldsymbol{0}$. We can further calculate that $Var(\boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{e}_j) = \boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{c} = \boldsymbol{0}$. Since $\boldsymbol{e}_j$ follows the multivariate Gaussian distribution with mean $\boldsymbol{0}$, $\boldsymbol{u}^T\boldsymbol{x}_j = \boldsymbol{c}^T\boldsymbol{B}_1\boldsymbol{e}_j = \boldsymbol{0}$. It follows that $\boldsymbol{u}^T\boldsymbol{X} = \boldsymbol{u}^T[\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n] = \boldsymbol{0}$, which contradicts our assumption that $\boldsymbol{X}$ has full rank. Thus, we can conclude the claim. ■

Lemma 2: Let $\widetilde{\boldsymbol{A}} = [\boldsymbol{A}, \boldsymbol{U}\boldsymbol{B}_1^T] \in \mathbb{R}^{m\times m}$. Then $rank(\widetilde{\boldsymbol{A}}) = m$.

Proof: This lemma can also be proved by contradiction. If $rank(\widetilde{\boldsymbol{A}}) < m$, then again there exists $\boldsymbol{c} \in \mathbb{R}^{m\times 1}$ such that $\boldsymbol{c}^T\widetilde{\boldsymbol{A}} = [\boldsymbol{c}^T\boldsymbol{A}, \boldsymbol{c}^T\boldsymbol{U}\boldsymbol{B}_1^T] = \boldsymbol{0}$. Note that $\boldsymbol{c}^T\boldsymbol{A} = \boldsymbol{0}$ implies $\boldsymbol{c} \in Nul(\boldsymbol{A}^T)$. Then there exists $\boldsymbol{c}'$ such that $\boldsymbol{c} = \boldsymbol{B}_1^T\boldsymbol{c}'$. It follows that $\boldsymbol{c}^T\boldsymbol{U}\boldsymbol{B}_1^T = \boldsymbol{c}'^T\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T = \boldsymbol{0}$. However, this contradicts Lemma 1, which completes the proof. ■

Based on lemma 1, matrix $\boldsymbol{B}_1\boldsymbol{D}\boldsymbol{B}_1^T$ has full rank and it is also symmetric. It means that we can find a $(m-k)\times(m-k)$ matrix $\boldsymbol{B}_2$, such that $\boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{B}_2^T = \boldsymbol{I}_{(m-k)\times(m-k)}$. Besides, according to Lemma 2, the inverse of the matrix $\widetilde{\boldsymbol{A}}$ uniquely exists. Let $\boldsymbol{B}_3 = [\boldsymbol{I}_{k\times k}, \boldsymbol{0}_{k\times(m-k)}]\widetilde{\boldsymbol{A}}^{-1}$ be the first k rows of $\widetilde{\boldsymbol{A}}^{-1}$. Then we have $\boldsymbol{B}_3\widetilde{\boldsymbol{A}} = [\boldsymbol{B}_3\boldsymbol{A}, \boldsymbol{B}_3\boldsymbol{U}\boldsymbol{B}_1^T] = [\boldsymbol{I}_{k\times k}, \boldsymbol{0}_{k\times(m-k)}]$. Finally, we claim that $\boldsymbol{W} = \begin{bmatrix}\boldsymbol{B}_3 \\ \boldsymbol{B}_2\boldsymbol{B}_1\end{bmatrix}$ satisfies the condition in the theorem. Specifically, for $j = 1,2,\cdots,n$,

$$\boldsymbol{W}\boldsymbol{x}_j = \begin{bmatrix}\boldsymbol{B}_3\boldsymbol{x}_j \\ \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{x}_j\end{bmatrix} = \begin{bmatrix}\boldsymbol{B}_3\boldsymbol{A}\boldsymbol{s}_j + \boldsymbol{B}_3\boldsymbol{e}_j \\ \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{A}\boldsymbol{s}_j + \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{e}_j\end{bmatrix} = \begin{bmatrix}\boldsymbol{s} + \boldsymbol{B}_3\boldsymbol{e}_j \\ \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{e}_j\end{bmatrix}, \quad (27)$$

because $\boldsymbol{B}_3\boldsymbol{A} = \boldsymbol{I}_{k\times k}$ and $\boldsymbol{B}_1\boldsymbol{A} = \boldsymbol{0}$. Since $\boldsymbol{e}$ follows the multivariate Gaussian distribution with mean $\boldsymbol{0}$, both $\boldsymbol{B}_3\boldsymbol{e}_j$ and $\boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{e}_j$ follow the Gaussian distribution with mean $\boldsymbol{0}$. In addition, we can calculate that

$$Var(\boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{e}_j) = \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{B}_2^T = \boldsymbol{I}_{(m-k)\times(m-k)}, \quad (28)$$

$$Cov(\boldsymbol{B}_3\boldsymbol{e}, \boldsymbol{B}_2\boldsymbol{B}_1\boldsymbol{e}_j) = \boldsymbol{B}_3\boldsymbol{U}\boldsymbol{B}_1^T\boldsymbol{B}_2^T = \boldsymbol{0}\boldsymbol{B}_2^T = \boldsymbol{0}. \quad (29)$$

Thus, the existence of $\boldsymbol{W}$ is proved by the construction.

APPENDIX B
DETAILED PROCEDURE OF THE NEWTON-LIKE METHOD ON SMOOTH MANIFOLD $SO(m)$

In this section, we present the details of the Newton-like method in one iteration. We mainly follow the steps in [16].

Let $\mathfrak{so}(m,\mathbb{R})$ denote the group of $m\times m$ skew-symmetric matrices, i.e., $\mathfrak{so}(m,\mathbb{R}) = \{\boldsymbol{\Omega} \in \mathbb{R}^{m\times m} | \boldsymbol{\Omega} = -\boldsymbol{\Omega}^T\}$. Then if $\boldsymbol{B} \in SO(m)$ and $\boldsymbol{\Omega} \in \mathfrak{so}(m,\mathbb{R})$, $\boldsymbol{B}\,exp(\boldsymbol{\Omega})$ also belongs to $SO(m)$. This is because $exp(\boldsymbol{\Omega})^T = exp(\boldsymbol{\Omega}^T) = exp(-\boldsymbol{\Omega}) = exp(\boldsymbol{\Omega})^{-1}$, and thus $\boldsymbol{B}\,exp(\boldsymbol{\Omega})\,[\boldsymbol{B}\,exp(\boldsymbol{\Omega})]^T = \boldsymbol{B}\,exp(\boldsymbol{\Omega})\,exp(\boldsymbol{\Omega})^T\boldsymbol{B}^T = \boldsymbol{B}\boldsymbol{B}^T = \boldsymbol{I}$. Besides, the group $\mathfrak{so}(m,\mathbb{R})$ is isomorphic to $\mathbb{R}^{m(m-1)/2}$. If we let ω_{ij} denote the ijth entry of a matrix $\boldsymbol{\Omega} \in \mathfrak{so}(m,\mathbb{R})$, then the vector $\bar{\boldsymbol{\Omega}} = [\omega_{ij}]_{1\le i<j\le m}$ has a length of $\frac{m(m-1)}{2}$, and $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}}$ can be uniquely determined by each other. For the vector $\bar{\boldsymbol{\Omega}}$, we only consider the value ω_{ij} when $i<j$, since when $i=j$, ω_{ij} must be 0 as $\boldsymbol{\Omega}$ is skew-symmetric. Therefore, the transformation $\mu_{\boldsymbol{B}}: \mathbb{R}^{m(m-1)/2} \to SO(m)$ can be defined as $\mu_{\boldsymbol{B}}(\bar{\boldsymbol{\Omega}}) = \boldsymbol{B}\,exp(\boldsymbol{\Omega})$. Obviously, we have $\mu_{\boldsymbol{B}}(\boldsymbol{0}) = \boldsymbol{B}\,exp(\boldsymbol{0}_{m\times m}) = \boldsymbol{B}$.

Before we consider the gradient and hessian matrix of $L\circ\mu_{\boldsymbol{B}}$ at $\boldsymbol{0}$, we first derive some auxiliary results. Let $\bar{\boldsymbol{\Omega}} = [\omega_{ij}]_{1\le i<j\le m}$ denote the direction we move in the space $\mathbb{R}^{m(m-1)/2}$ and ε denote the step size. Then the matrix $\boldsymbol{B}$ in the next iteration will be $\widetilde{\boldsymbol{B}} = \mu_{\boldsymbol{B}}(\varepsilon\bar{\boldsymbol{\Omega}}) = \boldsymbol{B}\,exp(\varepsilon\boldsymbol{\Omega})$. Therefore,

we can derive that

$$\left.\frac{d\widetilde{\boldsymbol{b}}_i}{d\varepsilon}\right|_{\varepsilon=0} = \left.\frac{d\widetilde{\boldsymbol{B}}}{d\varepsilon}\right|_{\varepsilon=0} \boldsymbol{e}_i = \boldsymbol{B}\boldsymbol{\Omega}\, exp(\varepsilon\boldsymbol{\Omega})|_{\varepsilon=0}\boldsymbol{e}_i = \boldsymbol{B}\boldsymbol{\Omega}\boldsymbol{e}_i = \boldsymbol{B}\boldsymbol{\omega}_i, \quad (30)$$

where $\widetilde{\boldsymbol{b}}_i$ is the ith column of $\widetilde{\boldsymbol{B}}$, $\boldsymbol{e}_i$ is the ith standard basis function where all the entries are 0 except the ith entry, which is 1. Furthermore,

$$\left.\frac{d^2\widetilde{\boldsymbol{b}}_i}{d\varepsilon^2}\right|_{\varepsilon=0} = \left.\frac{d\boldsymbol{B}\boldsymbol{\Omega}\, exp(\varepsilon\boldsymbol{\Omega})}{d\varepsilon}\right|_{\varepsilon=0} \boldsymbol{e}_i = \boldsymbol{B}\boldsymbol{\Omega}^2 exp(\varepsilon\boldsymbol{\Omega})|_{\varepsilon=0}\boldsymbol{e}_i = \boldsymbol{B}\boldsymbol{\Omega}\boldsymbol{\omega}_i \quad (31)$$

Next, let us consider the gradient of $L \circ \mu_{\boldsymbol{B}}$ at $\mathbf{0}$. Since directly calculating the derivative is difficult, we calculate the derivative of $L \circ \mu_{\boldsymbol{B}}$ with respect to the step size ε when the movement direction in $\mathbb{R}^{m(m-1)/2}$ is $\bar{\boldsymbol{\Omega}}$. We have

$$\begin{aligned}
&\left.\frac{d}{d\varepsilon}(L \circ \mu_{\boldsymbol{B}})(\varepsilon\bar{\boldsymbol{\Omega}})\right|_{\varepsilon=0} \\
&= \left.\sum_{i=1}^{m}\sum_{j_1=1}^{n} \frac{\sum_{j_2=1}^{n}\phi'\left(\frac{\widetilde{\boldsymbol{b}}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}{\sum_{j_2=1}^{n}\phi\left(\frac{\widetilde{\boldsymbol{b}}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)} \frac{(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T}{h}\frac{d\widetilde{\boldsymbol{b}}_i}{d\varepsilon}\right|_{\varepsilon=0} \\
&= \frac{1}{h}\sum_{i=1}^{m}\sum_{j_1=1}^{n} \frac{\sum_{j_2=1}^{n}\phi'\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}\boldsymbol{\omega}_i}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)} \\
&= \frac{1}{h}\sum_{i=1}^{m}\left[\sum_{j_1=1}^{n} \frac{\sum_{j_2=1}^{n}\phi'\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}\right]\boldsymbol{\omega}_i
\end{aligned} \quad (32)$$

Note that $\omega_{ij} = -\omega_{ji}$, if we denote τ_{ij} as

$$\tau_{ij} = \frac{1}{h}\sum_{j_1=1}^{n} \frac{\sum_{j_2=1}^{n}\phi'\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{b}_j}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}, \quad (33)$$

then we have

$$\begin{aligned}
\left.\frac{d}{d\varepsilon}(L \circ \mu_{\boldsymbol{B}})(\varepsilon\bar{\boldsymbol{\Omega}})\right|_{\varepsilon=0} &= \sum_{i=1}^{m}\sum_{j=1}^{m}\tau_{ij}\omega_{ji} \\
&= \sum_{1\le i<j\le m}\omega_{ij}(\tau_{ji}-\tau_{ij}),
\end{aligned} \quad (34)$$

which implies that the gradient of $L \circ \mu_{\boldsymbol{B}}$ at $\mathbf{0}$ is $[\tau_{ji} - \tau_{ij}]_{1\le i<j\le m}$.

In the next step, let us consider the Hessian matrix of $L \circ \mu_{\boldsymbol{B}}$ at $\mathbf{0}$. Similarly, we can calculate $\left.\frac{d^2}{d\varepsilon^2}(L \circ \mu_{\boldsymbol{B}})(\varepsilon\bar{\boldsymbol{\Omega}})\right|_{\varepsilon=0}$ as (35).

$$\begin{aligned}
&\left.\frac{d^2}{d\varepsilon^2}(L \circ \mu_{\boldsymbol{B}})(\varepsilon\bar{\boldsymbol{\Omega}})\right|_{\varepsilon=0} \\
&= \left.\sum_{i=1}^{m}\sum_{j_1=1}^{n}\frac{d}{d\varepsilon}\frac{\sum_{j_2=1}^{n}\phi'\left(\frac{\widetilde{\boldsymbol{b}}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}{\sum_{j_2=1}^{n}\phi\left(\frac{\widetilde{\boldsymbol{b}}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}\frac{(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}\boldsymbol{\Omega}\, exp(\varepsilon\boldsymbol{\Omega})\,\boldsymbol{e}_i}{h}\right|_{\varepsilon=0} \\
&= \underbrace{\frac{1}{h^2}\sum_{i=1}^{m}\sum_{j_1=1}^{n}\frac{\sum_{j_2=1}^{n}\phi''\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)\left[(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}\boldsymbol{\omega}_i\right]^2}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}}_{T_1} - \underbrace{\frac{1}{h^2}\sum_{i=1}^{m}\sum_{j_1=1}^{n}\frac{\left[\sum_{j_2=1}^{n}\phi'\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}\boldsymbol{\omega}_i\right]^2}{\left[\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)\right]^2}}_{T_2} \\
&\quad + \underbrace{\frac{1}{h}\sum_{i=1}^{m}\sum_{j_1=1}^{n}\frac{\sum_{j_2=1}^{n}\phi''\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{B}\boldsymbol{\Omega}\boldsymbol{\omega}_i}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}}_{T_3}
\end{aligned} \quad (35)$$

$$\kappa_{ll'}^{(1)} = \frac{1}{h^2}\sum_{j_1=1}^{n}\frac{\sum_{j_2=1}^{n}\phi''\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)\boldsymbol{b}_l^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})^T\boldsymbol{b}_{l'}^T}{\sum_{j_2=1}^{n}\phi\left(\frac{\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})}{h}\right)}, \quad (36)$$

Then we can consider these three terms T_1, T_2 and T_3 individually. Note that when $\boldsymbol{B}$ is close to the true unmixing matrix, $\boldsymbol{b}_i^T(\boldsymbol{z}_{j_1}-\boldsymbol{z}_{j_2})$ are independent for different i and have an expectation of 0. Thus, if we denote $\kappa_{ll'}^{(1)}$ as (36), $\kappa_{ll'} \approx 0$ if $l \neq l'$.

Besides,

$$T_1 = \sum_{i=1}^{m}\sum_{l=1}^{m}\sum_{l'=1}^{m}\omega_{li}\,\omega_{l'i}\kappa_{ll'}^{(1)} \approx \sum_{i=1}^{m}\sum_{l=1}^{m}\omega_{li}^2\kappa_{ll'}^{(1)}, \quad (37)$$

which implies that the term T_1 only contains the squared term of entries of $\mathbf{\Omega}$. Similarly, we can approximate the term T_2 and T_3 by summation of squared entries of $\mathbf{\Omega}$:

$$T_2 \approx \sum_{i=1}^{m}\sum_{l=1}^{m} \omega_{li}^2 \kappa_{ll'}^{(2)}, \qquad T_3 \approx \sum_{i=1}^{m}\sum_{l=1}^{m} \omega_{li}^2 \kappa_{ll'}^{(3)}, \tag{38}$$

If we define $\gamma_{ij} = \kappa_{ll'}^{(1)} + \kappa_{ll'}^{(2)} + \kappa_{ll'}^{(3)}$, it follows that

$$\begin{aligned} &\frac{d^2}{d\varepsilon^2}(L \circ \mu_{\boldsymbol{B}})(\varepsilon\bar{\mathbf{\Omega}})\Big|_{\varepsilon=0} \\ &= T_1 + T_2 + T_3 \\ &\approx \sum_{i=1}^{m}\sum_{j=1}^{m} \gamma_{ij}\omega_{ji}^2 \\ &= \sum_{1\le i<j\le m} \omega_{ij}^2(\gamma_{ji} + \gamma_{ij}) \end{aligned} \tag{39}$$

Consequently, the Hessian matrix of $L \circ \mu_{\boldsymbol{B}}$ at $\mathbf{0}$ can be approximated by a diagonal matrix with diagonal values $(\gamma_{ji} + \gamma_{ij})$ for $1 \le i < j \le m$. Finally, the updated point by Newton-like method in $\mathbb{R}^{m(m-1)/2}$ can be expressed as

$$\widetilde{\omega}_{ij} = \frac{\tau_{ji} - \tau_{ij}}{\gamma_{ji} + \gamma_{ij}}, 1 \le i < j \le m. \tag{40}$$

With the results above, we can iteratively update $\boldsymbol{B}$ until convergence is achieved (i.e., changes in $\boldsymbol{B}$ is negligible).

According to equations (32-40), the time complexity of each iteration is $O(m^2n^2)$, which is not high in general. However, when both m and n are extremely large, the computation may be slow. The generalization of this Newton-like method is discussed in Section VII.

REFERENCES

[1] C. Song and K. Liu, "Statistical degradation modeling and prognostics of multiple sensor signals via data fusion: A composite health index approach," *IISE Transactions*, vol. 50, no. 10, pp. 853–867, May 2018.

[2] M. Kim, C. Song, and K. Liu, "A Generic Health Index Approach for Multisensor Degradation Modeling and Sensor Selection," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 3, pp. 1426–1437, Jul. 2019.

[3] L. Yuan, Z. Zhou, Y. Yuan, and S. Wu, "An Improved FastICA Method for Fetal ECG Extraction," *Computational and Mathematical Methods in Medicine*, vol. 2018, pp. 1–7, 2018, doi: 10.1155/2018/7061456.

[4] M. McKeown, "Independent component analysis of functional MRI: what is signal and what is noise?," *Current Opinion in Neurobiology*, vol. 13, no. 5, pp. 620–629, Oct. 2003

[5] F. C. Viola et al. "Using ICA for the analysis of multi-channel EEG data." *Simultaneous EEG and fMRI: Recording, Analysis, and Application: Recording, Analysis, and Application*, pp. 121-133, 2010.

[6] Z. Ge and Z. Song, "Process Monitoring Based on Independent Component Analysis−Principal Component Analysis (ICA−PCA) and Similarity Factors," *Industrial & Engineering Chemistry Research*, vol. 46, no. 7, pp. 2054–2063, Mar. 2007.

[7] W. Zhou and R. B. Altman, "Data-driven human transcriptomic modules determined by independent component analysis," *BMC Bioinformatics*, vol. 19, no. 1, Sep. 2018.

[8] J. E, J. Ye, L. He, and H. Jin, "A denoising carbon price forecasting method based on the integration of kernel independent component analysis and least squares support vector regression," *Neurocomputing*, vol. 434, pp. 67–79, Apr. 2021.

[9] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Networks*, vol. 13, no. 4–5, pp. 411–430, Jun. 2000.

[10] J.-F. Cardoso, "Infomax and maximum likelihood for blind source separation," *IEEE Signal Processing Letters*, vol. 4, no. 4, pp. 112–114, Apr. 1997.

[11] J.-F. Cardoso, "High-Order Contrasts for Independent Component Analysis," *Neural Computation*, vol. 11, no. 1, pp. 157–192, Jan. 1999

[12] J.-F. Cardoso, "Blind signal separation: statistical principles," *Proceedings of the IEEE*, vol. 86, no. 10, pp. 2009–2025, 1998.

[13] R. Boscolo, H. Pan, and V. P. Roychowdhury, "Independent Component Analysis Based on Nonparametric Density Estimation," *IEEE Transactions on Neural Networks*, vol. 15, no. 1, pp. 55–65, Jan. 2004.

[14] Dinh Tuan Pham and P. Garat, "Blind separation of mixture of independent sources through a quasi-maximum likelihood approach," *IEEE Transactions on Signal Processing*, vol. 45, no. 7, pp. 1712–1725, Jul. 1997.

[15] R. J. Samworth and M. Yuan, "Independent component analysis via nonparametric maximum likelihood estimation," *The Annals of Statistics*, vol. 40, no. 6, Dec. 2012.

[16] H. Shen, S. Jegelka, and A. Gretton, "Fast Kernel-Based Independent Component Analysis," *IEEE Transactions on Signal Processing*, vol. 57, no. 9, pp. 3498–3511, Sep. 2009.

[17] A. Gretton, R. Herbrich, A. Smola, O. Bousquet, B. Schö, and lkopf, "Kernel Methods for Measuring Independence," *Journal of Machine Learning Research*, vol. 6, no. 70, pp. 2075–2129, 2005.

[18] F. R. Bach and M. I. Jordan, "Kernel Independent Component Analysis," *Journal of Machine Learning Research*, vol. 3, no. Jul, pp. 1–48, 2002.

[19] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[20] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," *Proceedings of the 24th international conference on Machine learning – ICML '07*, 2007.

[21] Y. Yao and G. Doretto, "Boosting for transfer learning with multiple sources," *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1855-1862, 2010.

[22] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain Adaptation via Transfer Component Analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[23] M. Long, J. Wang, G. Ding, J. Sun and P. S. Yu, "Transfer Feature Learning with Joint Distribution Adaptation," *2013 IEEE International Conference on Computer Vision*, pp. 2200-2207, 2013.

[24] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," *Proceedings of the 32nd International Conference on International Conference on Machine Learning* – Vol 37 (ICML'15), pp. 97–105. 2015.

[25] E. Tzeng, J. Hoffman, T. Darrell and K. Saenko, "Simultaneous Deep Transfer Across Domains and Tasks," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 4068-4076, 2015.

[26] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, Oct. 2018.

[27] H.-W. Ng, V. D. Nguyen, V. Vonikakis, and S. Winkler, "Deep Learning for Emotion Recognition on Small Datasets using Transfer Learning," *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction – ICMI '15*, 2015.

[28] N. Tajbakhsh *et al.*, "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?" *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1299–1312, May 2016.

[29] H. Jung, S. Lee, J. Yim, S. Park and J. Kim, "Joint Fine-Tuning in Deep Neural Networks for Facial Expression Recognition," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2983-2991, 2015.

[30] A. Tharwat, "Independent component analysis: An introduction," *Applied Computing and Informatics*, Aug. 2018, doi: 10.1016/j.aci.2018.

[31] M Ullsperger and S. Debener, *Simultaneous EEG and fMRI : recording, analysis, and application*. New York: Oxford University Press, 2010, pp. 122.

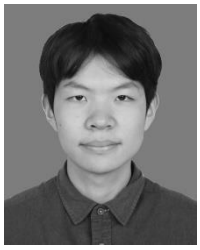
[32] D. Acharya and G. Panda, "A Review of Independent Component Analysis Techniques and their Applications," *IETE Technical Review*, vol. 25, no. 6, p. 320, 2008

[33] T. Hastie, and R. Tibshirani. "Independent components analysis through product density estimation." *Advances in neural information processing systems* 15, 2002.

[34] Hao Shen, S. Jegelka, and A. Gretton, "Fast Kernel-Based Independent Component Analysis," *IEEE Transactions on Signal Processing*, vol. 57, no. 9, pp. 3498–3511, Sep. 2009.

[35] B. W. Silverman, *Density estimation for statistics and data analysis*. Boca Raton: Chapman & Hall/Crc, 1998.

[36] K. Huper, and J. Trumpf. "Newton-like methods for numerical optimization on manifolds." *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers, 2004.*. Vol. 1. IEEE, 2004.
[37] F. Vrins and M. Verleysen, "On the entropy minimization of a linear mixture of variables for source separation," *Signal Processing*, vol. 85, no. 5, pp. 1029–1044, May 2005.
[38] S. Nikam and S. Deosarkar, "Fast ICA based technique for non-invasive fetal ECG extraction," *2016 Conference on Advances in Signal Processing (CASP)*, 2016, pp. 60-65, doi: 10.1109/CASP.2016.7746138.
[39] R. Vullings and J. O. E. H. van Laar, "Non-invasive Fetal Electrocardiography for Intrapartum Cardiotocography," Frontiers in Pediatrics, vol. 8, Dec. 2020, doi: 10.3389/fped.2020.599049.
[40] A. L. Goldberger *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet," *Circulation*, vol. 101, no. 23, Jun. 2000.
[41] J. Jezewski, A. Matonia, T. Kupka, D. Roj, and R. Czabanski, "Determination of fetal heart rate from abdominal signals: evaluation of beat-to-beat accuracy in relation to the direct fetal electrocardiogram," *Biomedizinische Technik/Biomedical Engineering*, vol. 57, no. 5, Jan. 2012, doi: 10.1515/bmt-2011-0130.
[42] K. Surya and K. K. Abdul Majeed, "Multichannel Probabilistic Framework for Prenatal Diagnosis of Fetal Arrhythmia Using ECG," *Transactions on Computational Science and Computational Intelligence*, pp. 141–150, 2021, doi: 10.1007/978-3-030-49500-8_13.
[43] R. Nurani, E. Chandraharan, V. Lowe, A. Ugwumadu, and S. Arulkumaran, "Misidentification of Maternal Heart Rate as Fetal on Cardiotocography During the Second Stage of Labor," Obstetrical & Gynecological Survey, vol. 68, no. 4, pp. 261–263, Apr. 2013.
[44] L. Dinh, D. Krueger, and Y. Bengio, "NICE: Non-linear Independent Components Estimation," *arXiv:1410.8516 [cs]*, Apr. 2015.
[45] A. Schell and H. Oberhauser, "Nonlinear Independent Component Analysis for Continuous-Time Signals," *arxiv.org*, Feb. 2021, Accessed: Feb. 22, 2022. [Online]. Available: https://arxiv.org/abs/2102.02876.



Ziqian Zheng received the B.E. degree in automation engineering from Xi'an Jiaotong University, Xi'an, China, in 2019. He is currently pursuing the Ph.D. degree with the Department of Industrial and Systems Engineering, University of Wisconsin–Madison, Madison, WI, USA.

His research interests are spatio-temporal modeling and anomaly detection.

Mr. Zheng is a member of the INFORMS, IISE, and SME.



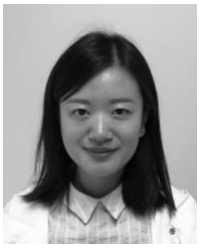
Wei Zhao received his MS degree in Computer Science from University of Wisconsin, Madison, WI, USA, in 2018 and Ph.D. degree in Intelligent transportation system. He has been a data scientist specialist with 3M, Maplewood, MN, USA since 2019. His current research interests include Smart manufactory and Deep Learning and Computer vision.



Brock Hable received his MS degree in Electrical Engineering from University of Minnesota, Minneapolis, MN, USA, in 2014 and BS degree in Electrical Engineering and Computer Science from University of Wisconsin, Madison, WI, USA in 2004. He has been a Researcher and Research Manager with 3M, Maplewood, MN, USA since 2004. His current research interests include Artificial Intelligence and Machine Learning and Edge Solutions for Industrial Applications.



Yutao Gong received the B.S. degree in Chemical Engineering from the University of Pittsburgh, Pittsburgh, PA, USA, in 2015, the M.S. degree in Statistics and the Ph.D. degree in Chemical Engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2019 and 2020, respectively. He currently works for 3M Company in the Corporate Research Process Laboratory as a data scientist. His research interests are machine learning and computer vision for process improvement.



Xuan Wang received her B.S. degree in Electrical Engineering from Northwestern Polytechnical University, Xi'an, China, in 2012. She received her M.S. and Ph.D. degrees in Electrical Engineering from the University of Wisconsin - Madison in 2016 and 2019. She has been a data scientist with 3M, Maplewood, MN, USA since 2020. Her research interests include computer vision, data mining, and machine learning solutions for industrial applications.



Robert W. Shannon received a BA in Psychology from the University of Minnesota, in 2003 and earned a PhD in Neuroscience from the University of Minnesota in 2012, where he used neuroimaging and visual perception tools to explore subconscious processing of emotional visual stimuli. His research interests have evolved in the decade since his PhD, from brains to computers, most recently into the field of natural language processing. He now works for 3M Company in the Health Information Systems division as a data scientist.



Kaibo Liu (M'14) received the B.S. degree in industrial engineering and engineering management from the Hong Kong University of Science and Technology in 2009 and the M.S. degree in statistics and the Ph.D. degree in industrial engineering from the Georgia Institute of Technology in 2011 and 2013, respectively.

He is currently an Associate Professor in the Department of Industrial and Systems Engineering, University of Wisconsin-Madison, Madison, WI, USA. His research interests are data fusion for process modeling, monitoring, diagnosis, prognostics, and decision making.

Dr. Liu is a member of the INFORMS, IEEE, IISE, SME, and ASQ.