

# Independent Component Analysis

Kittithat Krongchon<sup>1</sup> and Kannan Lu<sup>1</sup>

<sup>1</sup>*Department of Physics, University of Illinois  
at Urbana-Champaign, Urbana, IL 61801, USA*

(Dated: May 2, 2022)

# Abstract

Independent component analysis (ICA) is an unsupervised learning technique that is widely used in extracting independent factors in image, sound and medical signals. In this document, we review the basic notions of the ICA including mathematical formulations and detailed algorithms. We also implement different algorithms based on higher-order statistics and information theoretic approaches. These algorithms are applied to sound signal disentanglement and EEG blinking signal removal with success.

## INTRODUCTION

Independent component analysis (ICA) is one of the classical unsupervised learning techniques that is used to separate out latent variables or reduce dimensions. The motivation of the development of this technique is based on very common problems. For instance, in reality, source signals are often corrupted with 'noise'. Or in other words, data are composed by multiple independent source signals. To isolate the signals from different sources, ICA can be a good choice. The famous example is the 'cocktail party problem', where the sound detectors record the superposition of various sound signals from different sources in the party. The goal is to separate out the observed signal into sounds from each different source. Since we do not know what the sources are and how these signals are mixed, this type of problem is called blind source separation (BSS). ICA is widely used in images, sounds, stock market and medical signals. ICA can also be viewed as an extension of principal component analysis (PCA), where the latter maximizes the second order statistics (covariance matrix of data). ICA maximizes higher-order statistics or simply tries to look for independent components not just uncorrelated components. In Fig. 1, we show the ICA components  $IC_i$  and PCA components  $PC_i$  learned from the observed data  $x_i$  with the original sources  $s_i$  sampled from uniform distribution independently. Clearly, the ICA learned the independent components correctly but the PCA did not in this case.

In this paper, we will review mathematical background of the ICA technique, the detailed algorithm implementation and several realizations in typical examples.

## MATHEMATICAL BACKGROUND

### Definition of the problem and notations

Before we delve into the detailed mathematical formalisms, we first define the notations that we will use throughout the whole document. We will use boldface capital letters  $\mathbf{A}$  for matrices, boldface lowercase letters  $\mathbf{a}$  for vectors and ordinary lower case letters  $a$  for scalars. We will use subscripts for denoting the components of matrices (i.e.  $A_{ij}$ ) and vectors (i.e.  $a_i$ ). Notice that these letters are not in boldface as they mean the specific component, which is just a scalar in  $\mathbb{R}$ . Superscripts are reserved for indexing different samples. For instance,  $m$  samples of vector  $\mathbf{a}$  can be denoted as  $\mathbf{a}^{(k)}$  where  $k = 1, 2, \dots, m$ . Following these conventions,

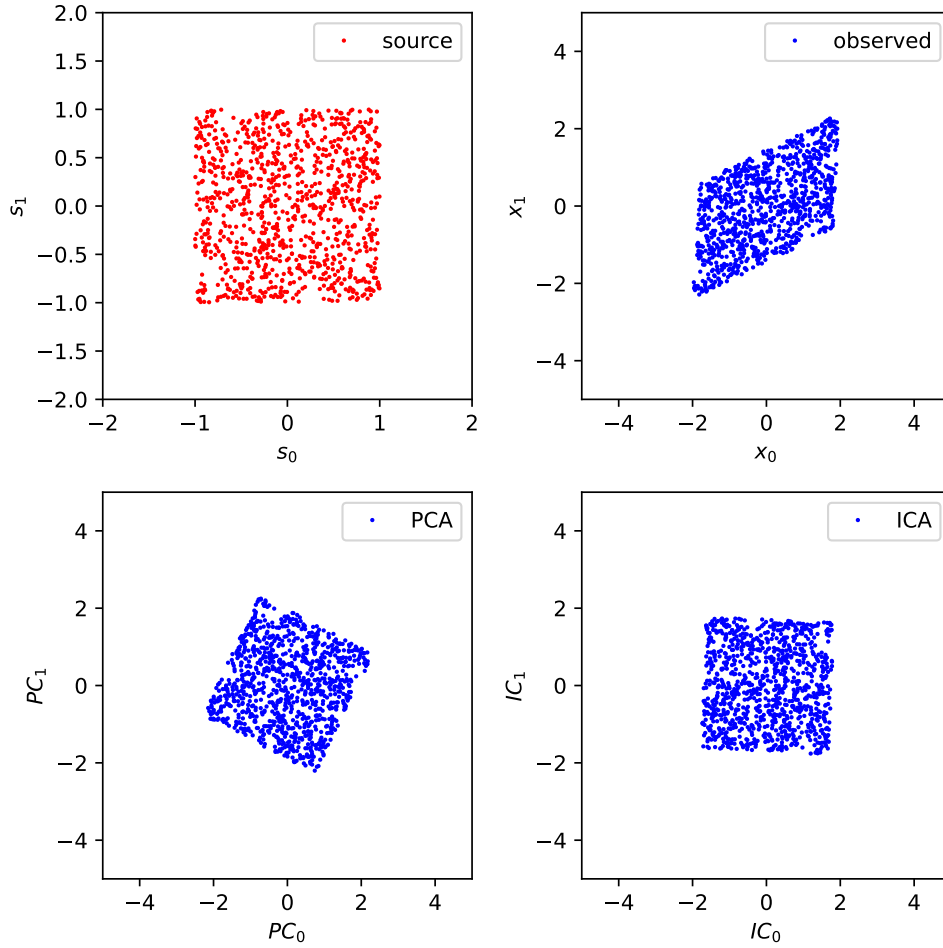


FIG. 1. shows the PCA and ICA result of unmixing the observed data  $x_i$  generated from two independent uniform variables  $s_i$ . The algorithm is based on maximizing the higher order moments.

we denote the original signal to be  $\mathbf{s}^{(i)} \in \mathbb{R}^d$ . We consider the standard setting where the  $d$  dimensional signals  $\mathbf{s}^{(i)} \in \mathbb{R}^d$  ( $d$  original sources) are mixed and observed by  $d$  detectors  $\mathbf{x}^{(i)} \in \mathbb{R}^d$  (i.e. the cocktail party scenario). That is there is a linear map between the  $\mathbf{s}^{(i)}$  and  $\mathbf{x}^{(i)}$ ,  $\mathbf{x}^{(i)} = \mathbf{A}\mathbf{s}^{(i)}$ , where  $\mathbf{A} \in \mathbb{R}^{d \times d}$  is known as the mixing matrix. The whole idea of ICA is to learn the inverse of the mixing matrix  $\mathbf{W} = \mathbf{A}^{-1}$  from the observed data  $\mathbf{x}^{(i)}$ , known as the un-mixing process. The learned independent components are denoted by  $\mathbf{u} = \mathbf{W}\mathbf{x}$ . This can be done in several different ways based on ideas of separating non-Gaussian signals, and the original signals being independent. We will in this section discuss various mathematical formulations. All these various theoretical frameworks are based on several constraints of the problem. These are:

- (1) The number of sensors is larger than number of sources. This ensures that the mixing matrix is full rank;
- (2) The sources at each sample (time) are mutually independent;
- (3) At most one source is normally distributed.

Without these conditions, the BSS problem is ill-defined. For simplicity, we will also restrict the discussion to cases where the number of detectors and sources are the same. However, a generalization to condition (1) is feasible and the algorithms discussed in the following sections will still work.

Based on these definitions, we need to talk about some ambiguities embedded in the symmetry of the problem and justify that if the original signals are all Gaussians, they cannot be learned through the un-mixing. There are two ambiguities that usually do not affect the practical application, namely the permutation ambiguity and the scaling ambiguity. It is trivial to see that the ordering of the original sources  $\{s_j\}$  is ambiguous. In the case of scaling, if the original signal  $\mathbf{s}$  is scaled by a non-zero constant to be  $c\mathbf{s}$  where  $c \neq 0$ . Then the mixing matrix  $\mathbf{A}$  can be scaled by  $1/c$  and resulting in the same observed data  $\mathbf{x} = \frac{1}{c}\mathbf{A}c\mathbf{s}$ . This scaling ambiguity can be further extended with regard to each component of the original signal. That is, for a particular component  $j$ , if we scale the component by  $c_j$  the corresponding column of the mixing matrix can be scaled by  $1/c_j$  to have the observed data unchanged (i.e.  $x_i = \sum_j \frac{1}{c_j} A_{ij} c_j s_j$ ).

The other ambiguity that matters for the practical application is that the original sources cannot all be distributed as Gaussians. Or to be more precise, in order to separate the independent components, we require that the original signals can only have at most one

component to be sampled from a Gaussian distribution, i.e.  $s_j \sim \mathcal{N}(\mu, \sigma^2)$  for at most one  $j$ . Let's consider the case when all the original independent signals are Gaussians. That is,  $s_j \sim \mathcal{N}(\mu_j, \sigma_j^2)$  for  $j = 1, 2, \dots, d$ . Then given the mixing matrix  $\mathbf{A}$ , we have

$$\mathbb{E}_{\mathbf{s}}[\mathbf{x}] = \mathbf{A}\mathbb{E}[\mathbf{s}] = \mathbf{A}\boldsymbol{\mu} \quad (1)$$

and

$$\text{Cov}[\mathbf{x}] = \mathbb{E}_{\mathbf{s}}[\mathbf{A}\mathbf{s}\mathbf{s}^t\mathbf{A}^t] - \mathbb{E}_{\mathbf{s}}[\mathbf{A}\mathbf{s}]\mathbb{E}[(\mathbf{A}\mathbf{s})^t] = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^t - \mathbf{A}\mathbf{M}\mathbf{A}^t \quad (2)$$

, where  $\mathbf{M}$  and  $\boldsymbol{\Sigma}$  are diagonal matrices of  $\mathbb{R}^{d \times d}$  with diagonal elements to be  $\{\mu_j^2\}$  and  $\{\sigma_j^2\}$ , respectively. Due to the scale ambiguity, this is the same as considering normally distributed signals with unity variance. Since each column  $j$  of the mixing matrix  $\mathbf{A}$  can be scaled by  $\sigma_j$ ,  $\tilde{\mathbf{A}}_{:,j} = \sigma_j \mathbf{A}_{:,j}$  and correspondingly the signal needs to be scaled by  $1/\sigma_j$ ,  $\tilde{s}_j = \frac{s_j - \mu_j}{\sigma_j}$ . Then,

$$\text{Cov}[\mathbf{x}] = \tilde{\mathbf{A}}\mathbf{I}_{d \times d}\tilde{\mathbf{A}}^t. \quad (3)$$

Now, if we consider a rotation  $\mathbf{R} \in O(d)$  acting on the scaled source signals  $\tilde{\mathbf{s}}$ . The mixing matrix then changes to  $\tilde{\mathbf{A}}\mathbf{R}$ . Upon this rotation, we would observe  $\mathbf{x}'$  as  $\mathbf{x}' = \tilde{\mathbf{A}}\mathbf{R}\tilde{\mathbf{s}}$ .  $\mathbf{x}'$  is again normally distributed and has covariance matrix,

$$\text{Cov}[\mathbf{x}'] = \tilde{\mathbf{A}}\mathbf{R}\mathbf{I}_{d \times d}\mathbf{R}^t\tilde{\mathbf{A}}^t = \tilde{\mathbf{A}}\mathbf{I}_{d \times d}\tilde{\mathbf{A}}^t. \quad (4)$$

This simply means that whether the sources are rotated or not the observed data will be distributed as  $\mathcal{N}(0, \tilde{\mathbf{A}}\tilde{\mathbf{A}}^t)$ . Thus, because of the rotational symmetry of the multivariate Gaussian distribution, we cannot separate and obtain the original source signals. These derivations also indicate that given the observed data  $\mathbf{x}$ , we can whiten the data and it will not affect the separated signals (up to scalings and mean translations).

### Higher order statistics approach

The non-Gaussianity thus directly provide us with the methodologies used in finding the independent component. Notice that the PCA only finds the uncorrelated components but not necessarily independent. The ICA tries to find independent components where the joint distributions can be factorized into marginal distributions by investigating into higher-order moments. There are multiple moment-based objective functions have been used along the

development of ICA. We will define some of the notations and stick to one of the moment-based objective functions in this paper.

As we discussed previously, we have the freedom to whiten the observed signals and we will denote the whitened observed data as  $\tilde{\mathbf{x}}$ , where each component is  $\tilde{x}_j = \frac{x_j - \mathbb{E}[x_j]}{\sigma_{x,j}}$ . The skewness, kurtosis and excess kurtosis are defined as

$$\gamma(x) := \mathbb{E}[\tilde{x}^3], \beta(x) := \mathbb{E}[\tilde{x}^4], \kappa(x) := \beta(x) - 3$$

. The objective function is a convex combination of the squared third and fourth cumulants defined as follows,

$$L_\alpha(\mathbf{w}) := \alpha \gamma^2(\mathbf{w}^\top \tilde{\mathbf{x}}) + (1 - \alpha) \kappa^2(\mathbf{w}^\top \tilde{\mathbf{x}}) \quad (5)$$

, where  $\alpha \in [0, 1]$  and  $\mathbf{w}^\top \mathbf{w} = 1$ .

### Information theoretic approach

Apart from the moment based approaches, there are several information theoretic formalisms. Here in this section, we will review all different information theoretic frameworks and show that essentially they are equivalent. Then we will talk about the learning rules for these theoretical formulations in the subsequent section.

The centralized quantity in all these formalisms is the mutual information, which is defined as the KL divergence of the multivariate distribution of the observed data  $\mathbf{x}$  and product of all its marginal distributions, i.e.

$$I(\mathbf{x}) = \sum_{\mathbf{x}} \mathbf{p}(\mathbf{x}) \log \frac{\mathbf{p}(\mathbf{x})}{\prod_i \mathbf{p}_i(\mathbf{x}_i)}. \quad (6)$$

$I(\mathbf{x})$  is non-negative and is 0 if and only if the  $\mathbf{x}$  are independent. Several different information theoretic approaches have been formulated in history and can be unified in the concept of minimizing the mutual information of the separated components  $\mathbf{u} = \mathbf{W}\mathbf{x}$ .

One of them is a key finding in neural networks, that is, maximizing information between inputs  $\mathbf{x}$  and outputs  $\mathbf{y}$  implies that the output distributions are factorized, thus minimizes the mutual information in the outputs  $\mathbf{y}$ . Maximizing the information between inputs and outputs is to maximize the output joint entropy  $H(\mathbf{y}) = \sum_i \mathbf{H}(\mathbf{y}_i) - \mathbf{I}(\mathbf{y})$ , where  $I(\mathbf{y}) = \sum_{\mathbf{y}} \mathbf{p}(\mathbf{y}) \log \frac{\mathbf{p}(\mathbf{y})}{\prod_i \mathbf{p}_i(\mathbf{y}_i)}$  is the mutual information in the outputs  $\mathbf{y}$ . The mutual information is non-negative and is zero if and only if  $\mathbf{y}$  are marginalized. Recall that in neural network, the

output is given by the nonlinearity  $y_i = g_i(\mathbf{w}_i^t \mathbf{x}) := g_i(u_i)$ , thus  $p(y_i) = \frac{p(u_i)}{|\frac{\partial g_i}{\partial u_i}|}$ . The maximum is therefore obtained by considering the gradient,

$$\frac{\partial H(\mathbf{y})}{\partial \mathbf{W}} = \frac{\partial(-I(\mathbf{y}))}{\partial \mathbf{W}} - \frac{\partial}{\partial \mathbf{W}} \sum_i \mathbb{E}[\log \frac{p(u_i)}{|\frac{\partial g_i}{\partial u_i}|}]. \quad (7)$$

This implies that the nonlinear function  $g_i$  in the neural network  $y_i = g(\mathbf{w}_i^t \mathbf{x})$  is a cdf of the source distribution  $s_i$  in order to kill the second term. Together with the constraint that the outputs are marginalized, this gradient is zero. Therefore, a good estimation of the cdf of the source signal improves drastically the performance of ICA. Notice that, if  $I(\mathbf{y}) = \mathbf{0}$ , the  $\mathbf{u} = \mathbf{W}\mathbf{x}$  should also satisfy  $I(\mathbf{u}) = \mathbf{0}$  as  $g_i$  is an invertible monotonic function. Hence, the neural network approach can be eventually reduced to minimizing the mutual information  $I(\mathbf{u})$ .

Another way to formulate this is to maximize the negentropy  $J(u_i)$ , which is the KL divergence  $D(p(u_i)||p_G(u_i))$  between  $p(u_i)$  and Gaussian distribution  $p_G(u_i)$  with the same mean and covariance as  $p(u_i)$ . Recall that the Gaussian distribution has maximum entropy constrained with the mean and covariance. The negentropy thus measures non-Gaussianity. Requiring that the  $\mathbf{u}$  can be factorized and decorrelated, the sum of negentropies can be

written as

$$\sum_i J(u_i) = \sum_i D(p(u_i)||p_G(u_i)) \quad (8)$$

$$= \sum_i p(u_i) \log \frac{p(u_i)}{p_G(u_i)} \quad (9)$$

$$= \sum_{\mathbf{u}} p(\mathbf{u}) \log \frac{\prod_i \mathbf{p}(\mathbf{u}_i)}{\prod_i \mathbf{p}_G(\mathbf{u}_i)} \quad (10)$$

$$= \sum_{\mathbf{u}} p(\mathbf{u}) \log \frac{\prod_i \mathbf{p}(\mathbf{u}_i)}{\mathbf{p}_G(\mathbf{u})} \quad (11)$$

$$= \sum_{\mathbf{u}} p(\mathbf{u}) \log \frac{\prod_i \mathbf{p}(\mathbf{u}_i)}{\mathbf{p}(\mathbf{u})} + \sum_{\mathbf{u}} \mathbf{p}(\mathbf{u}) \log \frac{\mathbf{p}(\mathbf{u})}{\mathbf{p}_G(\mathbf{u})} \quad (12)$$

$$= D(\prod_i p(u_i)||p(\mathbf{u})) + \mathbf{J}(\mathbf{u}) \quad (13)$$

$$= -I(\mathbf{u}) + \mathbf{J}(\mathbf{u}) \quad (14)$$

$$= -I(\mathbf{u}) - \mathbf{H}(\mathbf{u}) - \sum_{\mathbf{u}} \mathbf{p}(\mathbf{u}) \log \mathbf{p}_G(\mathbf{u}) \quad (15)$$

$$= -I(\mathbf{u}) - \mathbf{H}(\mathbf{x}) - \log(|\det(\mathbf{W})|) - \frac{1}{2} \log((2\pi\mathbf{e})^d \det(\langle \mathbf{u}, \mathbf{u}^t \rangle)) \quad (16)$$

$$= -I(\mathbf{u}) - \mathbf{H}(\mathbf{x}) - \frac{1}{2} \log((2\pi\mathbf{e})^d). \quad (17)$$

In the derivation, the  $\mathbf{u}$  are uncorrelated so the covariance matrix is identity. Therefore, maximizing the negentropy is equivalent to minimizing mutual information in  $\mathbf{u} = \mathbf{W}\mathbf{x}$ .

Lastly, for the maximum likelihood estimation approach, we want to maximize the log likelihood over all samples observed upon choosing a parametrized distribution  $\hat{p}_s(\mathbf{w}_i\mathbf{x})$  satisfying  $p(\mathbf{x}) = \prod_i^d \hat{\mathbf{p}}_s(\mathbf{w}_i\mathbf{x})|\mathbf{W}|$ . The log-likelihood is

$$l(\mathbf{W}) = \frac{1}{N} \sum_{j=1}^N \left( \sum_{i=1}^d \log(\hat{\mathbf{p}}_s(\mathbf{w}_i\mathbf{x}^{(j)})) + \log |\mathbf{W}| \right). \quad (18)$$

If the approximation  $\hat{p}_s(\mathbf{w}_i\mathbf{x})$  are close to the actual pdf, the first term is approximately  $-\sum_i H(\mathbf{w}_i\mathbf{x})$  and is equal to the negative of the mutual information up to an additive constant of the total entropy of  $\mathbf{x}$ . Hence, all these historical information theoretic approaches are equivalent and the key idea is to approximate the probability distribution of the sources correctly. Historically, there has been several proposed parametric distributions for approximating super-Gaussian or sub-Gaussian distributions. We will proceed to discuss the algorithm associated with the information based approach in the next section.



## ALGORITHMS

In this section, we will discuss some of the common algorithms based on the previous mathematical formalisms. We will categorize them into moment-based approaches and information-based approaches. In either case, the problem is equivalent to an optimization problem. There are some detailed differences on how this optimization problem is implemented, i.e. ordinary gradient descent, Newton's method or fixed point algorithm. We will simply specify what we implemented for various examples that we will discuss later.

### Moment-based approach

In this method, we maximize the non-Gaussianity given by

$$J(Y) \propto \{E[G(Y)] - E[G(V)]\}^2, \quad (19)$$

where  $Y$  is a random variable of zero mean and unit variance, which can be achieved by whitening the data, and  $V$  is a Gaussian variable also of zero mean and unit variance. The function forms of  $G$  should not grow too fast to be robust. The following choice of  $G$  is proposed by Hyvärinen and Oja [1].

$$G(u) = \frac{1}{a_1} \log(\cosh a_1 u). \quad (20)$$

$$\partial_u G(u) = \tanh(a_1 u). \quad (21)$$

We want to find the extrema of  $E[G(Y)]$  to maximize  $J(Y)$  under the constraints

$$E[(w^T x)^2] = \|w\|^2 = 1. \quad (22)$$

From the Kuhn–Tucker conditions, the extremum condition is satisfied when

$$E[xg(w^T x)] - \beta w = 0. \quad (23)$$

This equation can be solved by using Newton's method. Let  $F(w)$  denote the left-hand side of the equation, which we are trying to solve. We update the value of  $w$  in each iteration

according to the following equation.

$$w_{n+1} = w_n - \frac{F(w_n)}{F'(w_n)} \quad (24)$$

$$= w_n - \frac{E[xg(w_n^T x)] - \beta w_n}{E[g'(w_n^T x)] - \beta} \quad (25)$$

$$= \frac{w_n E[g'(w_n^T x)] - \beta w_n - E[xg(w_n^T x)] + \beta w_n}{E[g'(w_n^T x)] - \beta} \quad (26)$$

$$= \frac{w_n E[g'(w_n^T x)] - E[xg(w_n^T x)]}{E[g'(w_n^T x)] - \beta}. \quad (27)$$

### Information theoretic approach

Based on what has been discussed in the previous section, we can approximate the source signal distribution with a parametrized pdf. There are multiple choices here tailored to different problems. A default choice is to assume the cdf of the source signal can be approximated by a sigmoid function  $g(s_i) = \frac{1}{1+e^{-s_i}}$ , thus the pdf is  $p(s_i) = g'(s_i)$ . Notice that this pdf is super-Gaussian. This means that ICA based on this pdf works well when the distribution of original sources have heavy tails and sharp peak around center. Recall the log-likelihood (without normalization),

$$l(\mathbf{W}) = \sum_{j=1}^N \left( \sum_{i=1}^d \log(\hat{p}_s(\mathbf{w}_i \mathbf{x}^{(j)})) + \log |\mathbf{W}| \right) \quad (28)$$

$$= \sum_{j=1}^N \left( \sum_{i=1}^d \log(g'(\mathbf{w}_i \mathbf{x}^{(j)})) + \log |\mathbf{W}| \right) \quad (29)$$

## APPLICATIONS

We implement FastICA according to the formalism described in the previous section and apply it to the observed data  $X$  in order to find the source matrix  $S$  given by

$$S = \begin{bmatrix} s_1(t_1) & s_1(t_2) & \cdots & s_1(t_N) \\ s_2(t_1) & s_2(t_2) & \cdots & s_2(t_N) \\ \vdots & \vdots & \ddots & \vdots \\ s_d(t_1) & s_d(t_2) & \cdots & s_d(t_N) \end{bmatrix}, \quad (30)$$

where  $N$  is the number of time steps,  $d$  is the number of detectors, and  $s_i$ 's are independent signal sources.

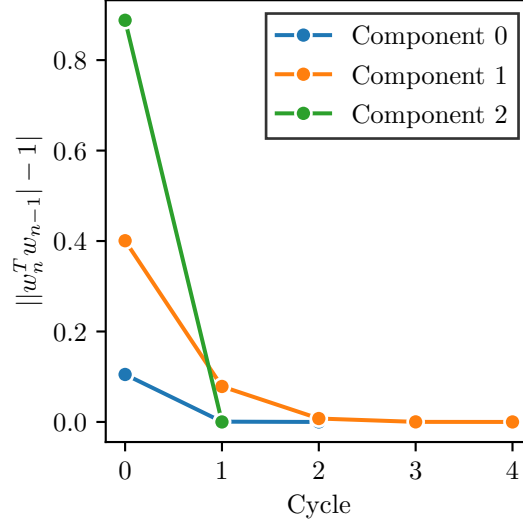


FIG. 2. The convergence criterion  $||w_n^T w_{n-1}| - 1|$  as a function of cycle for each row of  $W$ .

### Application 1: sine, square, and sawtooth waves

In this first example, we define the independent signal sources according to the following equations and plot them in Fig. 3(a).

$$\begin{aligned}
 s_1 &= \sin(2t). \\
 s_2 &= 2 \operatorname{sign}[\sin(3t)]. \\
 s_3 &= 4 \left[ \frac{1}{\pi} (2\pi t \bmod 2\pi) - 1 \right].
 \end{aligned} \tag{31}$$

The mixture signals  $X$  (Fig. 3(b)) that we want to apply FastICA on is constructed by the equation

$$X = AS, \tag{32}$$

where  $A$  is the mixing matrix, which in this example, is set to be

$$A = \begin{bmatrix} 1 & 1 & 1 \\ \frac{1}{2} & 2 & 1 \\ \frac{3}{2} & 1 & 2 \end{bmatrix}. \tag{33}$$

Newton's method converges rather quickly within only four cycles for each row of  $W$  as shown in Fig. 2. The sources predicted by the algorithm as implemented in our code and

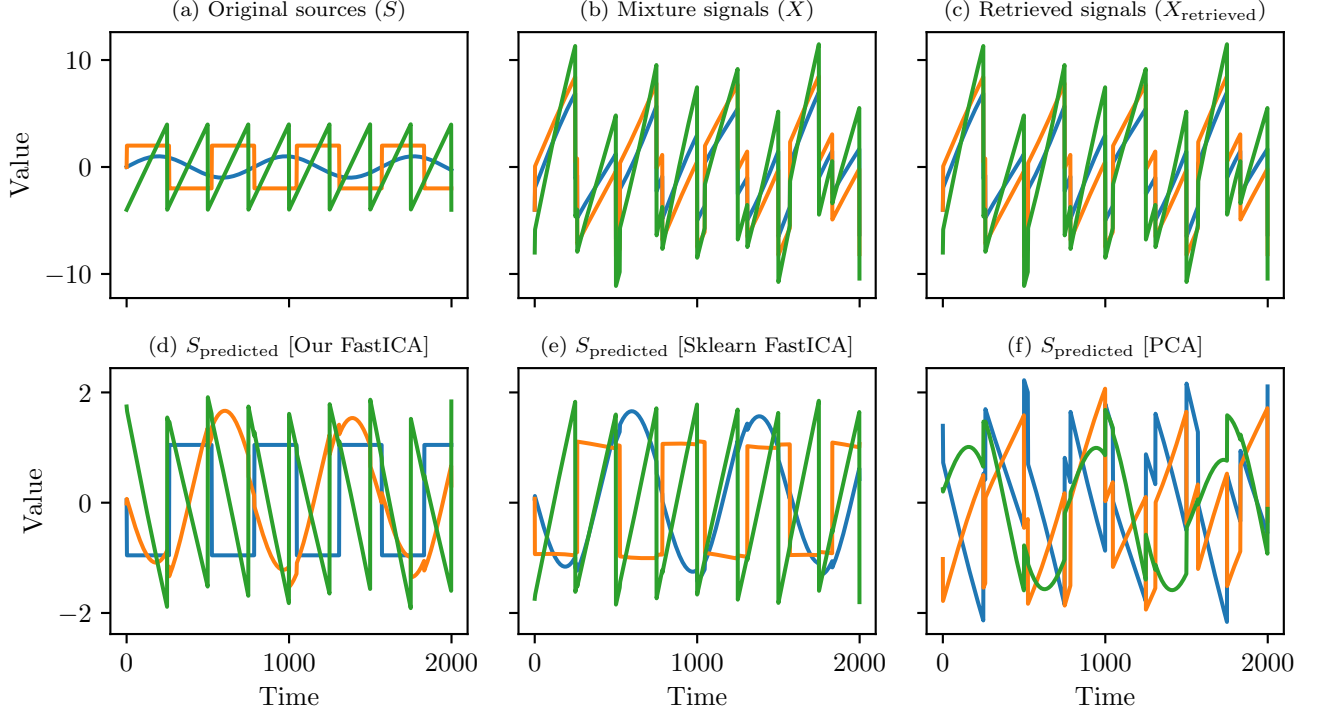


FIG. 3. (a) Original sources as defined in Eq. (31). (b) Mixture signals whose mixing matrix is defined by Eq. (33). (c) Retrieved input signals from the predicted sources according to Eq. (34). (d) Separated sources using our implementation of FastICA. (e) Separated sources using `sklearn.decomposition.FastICA`. (f) Separated sources using `sklearn.decomposition.PCA`.

in `sklearn.decomposition.FastICA` are shown in Fig. 3(d, e). Both of the results are very similar in terms of the ability to separate out independent sources. The main difference is the sign of the sawtooth wave because independence is up to a minus sign as discussed previously in the formalism section. From the predicted sources  $S_{\text{predicted}}$ , we can verify the accuracy of FastICA results by inverting the mixing and whitening transformations and adding back the mean of each mixture to retrieve the input signals.

$$(X_{\text{retrived}})_i = [(WK)^{-1}S_{\text{predicted}}]_i + E[X_i], \quad (34)$$

where  $K$  is the whitening transformation given by... The retrieved signals  $X_{\text{retrieved}}$  is shown to be equal to the original input signal  $X$  (Fig. 3(c)), which confirms that we can exactly reproduce the observed data from the output of ICA. In some applications, such as separation of independent sound sources, we might not need to retrieve the original signal. However, this procedure becomes crucial in other applications, such as blink removals of the

electroencephalogram data. The discussion of both mentioned applications follows.

In this specific example, the result from `sklearn.decomposition.PCA` is also plotted in Fig. 3(f). The figure shows that PCA is not suitable for separating independent sources.

## **Application 2: audio source separation**

## **HOW TO CHECK THE ROBUSTNESS AND CONVERGENCE**

$$WA = I. \tag{35}$$

## **WHAT IF TWO OF THE SOURCES ARE DEPENDENT**

## **DISCUSSIONS AND CONCLUSIONS**

1) review the traditional development from 1990s 2) recent advances and development from 2000s 3) what happens when there are multiple Gaussian components

---

- [1] A. Hyvärinen and E. Oja, Independent component analysis: algorithms and applications, Neural networks **13**, 411 (2000).