Boosting Independent Component Analysis

YunPeng Li ZhaoHui Ye

Abstract—Independent component analysis is intended to recover the mutually independent components from their linear mixtures. This technique has been widely used in many fields, such as data analysis, signal processing, and machine learning. To alleviate the dependency on prior knowledge concerning unknown sources, many nonparametric methods have been proposed. In this paper, we present a novel boosting-based algorithm for independent component analysis. Our algorithm consists of maximizing likelihood estimation via boosting and seeking unmixing matrix by the fixed-point method. A variety of experiments validate its performance compared with many of the presently known algorithms.

Index Terms—Independent component analysis, boosting, nonparametric maximum likelihood estimation, fixed-point method.

I. INTRODUCTION

INDEPENDENT Component Analysis (ICA) has received much attention in recent years due to its effective methodology for various problems, such as feature extraction, blind source separation, and exploratory data analysis. In the ICA model, a m dimensional random vector

\[ x = As \]

(1)

where s contains the mutually independent components, and A is called the mixing matrix. An unmixing matrix W is used to determine the estimated sources

\[ y = Wy \]

(2)

Without loss of generality, several assumptions are clear in ICA: (i) \( E(s) = 0 \) and \( Cov(s) = I \); (ii) each \( s_i \) is independent distributed and at most one of the sources is Gaussian \( i \); (iii) both A and W are invertible matrices. Since centering and whitening are often performed as preprocessing steps in the context of linear ICA, W is restricted to be an orthonormal matrix \( WW^T = I \).

ICA was firstly introduced to the neural network domain in the 1980s [2]. It was not until 1994 [1] that the theory of ICA was established. The most popular method is to optimize some contrast functions to achieve source separation. These contrast functions were usually chosen to represent the measure of independence or non-Gaussianity, for example, the mutual information [3], [4], the maximum entropy (or negentropy) [5]–[7], and the nonlinear decorrelation [8], [9].

In addition, higher-order moments methods [10], [11] were also designed to estimate the unknown sources. It has been pointed out that these contrast functions are related to the sources’ density distributions in the parametric maximum likelihood estimation [12]–[15]. These parametric methods’ performance are dependent on the choices of contrast functions or prior assumptions on the unknown sources’ distributions. Although it [5] has been shown that small misspecifications (for parametric methods) in the source densities do not affect local consistency of the estimation concerning the unmixing matrix W. Amari [16] demonstrated that ICA methods have to estimate the sources’ densities (nonparametric methods) to achieve the information bound concerning W.

Nonparametric ICA’s in-depth analysis and asymptotic efficiency were firstly available in [17]. There are currently two kinds of nonparametric ICA methods: the restriction methods and the regularization methods. In restriction methods, each source \( s_i \) belongs to certain density family or owns special structure, such as Gaussian mixtures models [18], [19], kernel density distributions [20], [21], and log-concave family [22]. The regularization methods determine the unknown sources via maximizing a penalized likelihood [23]. For other recent ICA approaches, see also [24]–[29].

Recently, we have successfully applied boosting to the nonparametric maximum likelihood estimation (boosting NPMLE) [30]. In this paper, we further propose a novel ICA method called BoostingICA based on our previous work. The proposed approach adaptively includes only those basis functions that contribute significantly to the sources’ estimation. Real data experiments validate its competitive performance with other popular or recent ICA methods.

II. NONPARAMETRIC MAXIMUM LIKELIHOOD ESTIMATION IN ICA

In this section, nonparametric maximum likelihood estimation is applied to solve the ICA problem. Maximizing the likelihood can be viewed as a joint maximization over the unmixing matrix W and the estimated sources’ densities, fixing one argument and maximizing over the other.

Let \( p_i(y_i) \) be the estimated density distribution for single component \( y_i \). Owing to the independence among \( y_i \), the estimated sources’ density distribution \( p_y(y) \) can be written as

\[ p_y(y) = \prod_{i=1}^{m} p_i(y_i) \]

(3)

Since there is a linear transform in equation (2), the joint density of the observed mixtures \( x \) is

\[ p_x(x; W, y) = |\text{det}(W)| p_y(Wx) = \prod_{i=1}^{m} p_i(w_i^T x) = \prod_{i=1}^{m} p_i(y_i) \]

(4)
where \( w^T_l \) is the \( i_{th} \) row in the orthonormal matrix \( W \) and \( |\text{det}(W)| = 1 \).

For parametric ICA, we have to determine the density distribution \( p_i(y_i) \) based on the prior assumptions beforehand, which might result in the unwanted density mismatching. To keep the density distribution of \( y_i \) unspecified, nonparametric methods are demanded. In the proposed method, we choose to model the estimated source \( y_i \) with Gibbs distribution in equation \( (5) \).

\[
p_i(y_i) = \frac{e^{f_i(y_i)}}{\int e^{f_i(y_i)} dy_i}
\]

where \( f_i(y_i) \) is assumed to be a smooth function in \( \mathbb{R} \), and the denominator of equation \( (5) \) is the partition function. Given \( N \) independent identically distributed samples \( \{x_j\}_1^N \), we can estimate the source’s sample \( y_i^q = w^T_l x_j \).

To simplify the integral in equation \( (5) \), we construct a grid of \( L \) values \( y_i^q \) (ascending order) with \( \Delta_i \), step, and let the corresponding frequency \( q_i^q \) be

\[
q_i^q = \sum_{j=1}^{N} \mathbb{I}(y_i^q \in (y_i^q - \Delta_i/2, y_i^q + \Delta_i/2)) / N
\]

where \( \mathbb{I}(\cdot) \) is the indicator function. The support of \( p_i(y_i) \) is restricted in \( [y_i^1 - \Delta_i/2, y_i^L + \Delta_i/2] \), and \( p_i(y_i) \) is then transformed as

\[
p_i(y_i) = e^{f_i(y_i)} / \sum_{i=1}^{L} \Delta_i e^{f_i(y_i)}/N
\]

It \([30], [31]\) has been shown that the log-likelihood \( \mathcal{L}(w_i, f_i) \) concerning \( y_i \) can be simplified to the following modified form

\[
\mathcal{L}(w_i, f_i) = \sum_{i=1}^{L} q_k^q \ln p_i(y_i^q) = \sum_{i=1}^{L} q_k^q f_i(y_i^q) - \Delta_i e^{f_i(y_i^q)}/N
\]

and the maximum log-likelihood is obtained when the partition function \( \sum_{i=1}^{L} \Delta_i e^{f_i(y_i^q)} = 1 \). Thus, the total modified log-likelihood in our method is

\[
\mathcal{L}(\{w_i, f_i\}_{i=1}^{m}) = \sum_{i=1}^{m} \mathcal{L}(w_i, f_i)
\]

The key of BoostingICA is to optimize equation \( (9) \) until convergence by joint maximization \([22], [23], [32]\), and the two iterative stages are illustrated in Algorithm \( 1 \). Similar joint maximization has been used in past researches concerning both projection pursuit \([33], [34]\) and ICA \([22], [23]\).

### A. Estimating the source’s density via boosting

To apply the boosting principle (forward stagewise regression \([35], [36]\) to nonparametric maximum likelihood estimation \([30]\), we regard \( f_i(y_i) \) in equation \( (5) \) as a combination of weak learners \( b_i(y_i; \gamma_{i,k}) \),

\[
f_i(y_i) = \sum_{k=1}^{M} b_i(y_i; \gamma_{i,k})
\]

where \( M \) is the number of boosting iterations and \( k \) is the index. Each single weak learner \( b_i(y_i; \gamma_{i,k}) \) is characterized by a set of parameters \( \gamma_{i,k} \) and is trained on the weighted data at the \( k_{th} \) iteration. We define the \( f_i \) at the \( k_{th} \) iteration as \( f_i^k \)

\[
f_i^k(y_i) = \sum_{r=1}^{k} b(y_i; \gamma_{i,r}) = f_i^{k-1}(x) + b(y_i; \gamma_{i,k})
\]

and the subproblem based on the former \( f_i^{k-1} \) becomes

\[
\max_{\gamma_{i,k}} \mathcal{L}(f_i^{k-1}(y_i) + b(y_i; \gamma_{i,k}))
\]

We maximize equation \( (12) \) via second-order approximation around \( f_i^{k-1} \), and the corresponding iterative reweighted least squares (IRLS) \([37]\) (Line 14) and updating strategy for the next iteration (Line 10-11) are shown in Algorithm \( 2 \). A penalty term \( \lambda J(b_i(y_i; \gamma_{i,k})) \) (Line 14) is added to the original least squares to restrict the model complexity of \( b_i(y_i; \gamma_{i,k}) \), where \( \lambda \) is the Lagrange multiplier and \( J(b_i(y_i; \gamma_{i,k})) \) is a nonnegative function. \( \omega_i^k \) (Line 10-11) is the weight and response of \( y_i^q \) at the \( k_{th} \) iteration, and they are simultaneously updated in the next iteration. Once all the weak learners have been trained, \( f_i(y_i) \) is the combination of whole \( M \) weak learners.

Inspired by the past researches \([23], [30]\), smooth spline is chosen as the weak learner for two reasons: Line 14 in Algorithm \( 2 \) can be efficiently computed in \( O(L) \) time \([23]\), and then the first and second derivatives of smooth spline (concerning \( y_i \)) are immediately available \([23]\). The penalty term is defined as \( J(b_i(y_i; \gamma_{i,k})) = \int \left[ b''_i(y_i; \gamma_{i,k}) \right]^2 dy_i \) (roughness penalty penalizes the curvature of function), and the degree of freedom \([39]\) \( df \) (corresponding to the prechosen \( \lambda \)) is defined implicitly by the trace of the linear smoother in Algorithm \( 2 \) (Line 14). With \( df \)'s increase from 2 to \( L \) (corresponding to \( \lambda \)'s decrease from \( \infty \) to 0), \( b_i(y_i; \gamma_{i,k}) \) changes from a simple line fit to the ordinary least squares fit. For single weak learner, we only need to determine its hyper-parameter \( df \) beforehand, simply choosing the small value (slightly greater than 2, and our method’s default value is 3) for \( df \) to restrict the model complexity (slightly more complex than the simple line fit).

---

**Algorithm 1:** BoostingICA algorithm

1: Initialization
2: maximum iterations \( \text{maxit} \)
3: grid value \( L \)
4: boosting iterations \( M \)
5: degree of freedom \( df \)
6: for \( it = 1 \) to \( \text{maxit} \) do
7: for \( i = 1 \) to \( m \) do
8: fixing \( W \), each \( f_i \) is estimated by boosting NPMLE (Algorithm \( 2 \)).
9: end for
10: given \( f_i \), \( W \) is restricted to be orthonormal and calculated via the fixed-point method (Algorithm \( 3 \)).
11: end for
12: Output
13: unmixing matrix \( W \)
14: estimated densities \( \{f_i\}_{i=1}^{m} \)
Algorithm 2 Estimating the density distribution of \( y_i \) by boosting NPMLE

1: Initialization
2: \( b_i(y_i; \gamma_{i,0}) \leftarrow -\frac{1}{2} y_i^2 - \frac{1}{2} \ln 2\pi \)
3: \( f_i^0(y_i) \leftarrow b_i(y_i; \gamma_{i,0}) \)
4: \( f_i^0(y_i) \leftarrow b_i(y_i; \gamma_{i,0}) \)
5: \( f_i^0(y_i) \leftarrow b_i(y_i; \gamma_{i,0}) \)
6: compute the \( L \) grid values \( \{y_{ij}\}_{i=1}^L \) and \( \Delta_i \) from \( \{y_{ij}\}_{j=1}^N \).
7: \( \omega_i^0 \leftarrow \Delta_i \frac{1}{\sqrt{2\pi}} e^{-\frac{y_{i,1}^2}{2}} \), \( 1 \leq i \leq L \)
8: for \( k = 1 \) to \( M \) do
9: for \( l = 1 \) to \( L \) do
10: \( \omega_i^k \leftarrow \omega_i^{k-1} e^{b_i(y_{ij}; \gamma_{i,k-1})} \)
11: \( y_{ij}^k \leftarrow \frac{y_{ij} - \omega_i}{\omega_i} \)
12: end for
13: solve the reweighted least squares with respect to \( \gamma_{i,k} \)
14: \( \sum_{i=1}^{L} \frac{1}{2} \omega_i^k (b_i(y_{ij}^k; \gamma_{i,k}) - Y_{ij}^k)^2 + \lambda J(b_i(y_i; \gamma_{i,k})) \)
15: \( f_i^k(y_i) \leftarrow f_i^{k-1}(y_i) + b_i(y_i; \gamma_{i,k}) \)
16: \( f_i^k(y_i) \leftarrow f_i^{k-1}(y_i) + b_i(y_i; \gamma_{i,k}) \)
17: \( f_i^k(y_i) \leftarrow f_i^{k-1}(y_i) + b_i(y_i; \gamma_{i,k}) \)
18: end for
19: Output
20: \( f_i(y_i) \leftarrow f_i^M(y_i) \)
21: \( f_i^p(y_i) \leftarrow f_i^M(y_i) \)
22: \( f_i^p(y_i) \leftarrow f_i^M(y_i) \)

In fact, when \( df \) is fixed with the default value, the number of boosting iterations \( M \) is the only hyper-parameter in the proposed method. Fortunately, it [40] has been shown that even when the training error is close to zero (log-likelihood is close to the upper-bound in our algorithm), boosting provably tends to increase the performance on the testing data with the increase of \( M \). We later provide robust experiment to validate this surprising phenomenon.

Figure 1 shows the estimated log-likelihood, and the true rotate angles (concerning the mixing matrix \( A \)) of mixtures \( x \) are plotted for comparison. Smooth spline surprisingly performs well when \( M = 1 \), and it successfully finds the ground-truth in two cases as the increase of boosting iterations \( M \).

B. Fixed-point method for estimating the separation matrix

Given the fixed \( f_i \), partition function becomes constant. Since there exists a whitening preprocessing stage for the observation \( x \) and the unmixing matrix \( W \) is orthonormal, fixed-point method developed in FastICA [5] can be used in our algorithm. Then, the log-likelihood is maximized by an approximative Newton method with quadratic convergence (see Algorithm 3).

C. Discussions of BoostingICA

Since there is a trade-off between \( df \) and \( M \) in boosting, simple hyper-parameters tuning strategies are available:

- decreasing the value of boosting iterations \( M \) might lead to the reduce of elapsed time, at the cost of weakening the density estimation for unknown sources.
- slightly increasing the weak learner’s model complexity \( df \) might benefit the density estimation without sacrificing time efficiency.
- we can level up the separation performance by slightly increasing \( df \), and reduce the elapsed time via cutting down \( M \).

Owing that \( s_i \) is zero mean and unit variance, the most uncertain (or circumspect) initialization (Line 2) in Algorithm [3] is the logarithm of the standard Gaussian, which is different from the zero initialization in the original boosting NPMLE [30].

III. EXPERIMENTS AND RESULTS

A. Implementation details

Our experiments were conducted on the R development platform with Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, and we intend to illustrate the proposed method’s comparable performance with other popular or recent ICA algorithms.

ICA algorithms used in our experiments are shown in Table 1. FixNA2 is the recent blind source separation algorithms based on nonlinear auto-correlation, and EFICA [41] is a statistically efficient version of the FastICA. All methods share the same maximum iterations \( maxit = 20 \), and EFICA’s elapsed time (with superscript *) was measured on the Matlab platform while other methods were R implementations.
The separation performance is measured by the value of Amari metrics $d(W, W_0)$ [42],
\[
d(W, W_0) = \frac{1}{2m} \sum_{i=1}^{m} \left( \frac{\sum_{j=1}^{n} |r_{ij}|}{\max_j |r_{ij}|} - 1 \right) + \frac{1}{2m} \sum_{j=1}^{n} \left( \frac{\sum_{i=1}^{m} |r_{ij}|}{\max_i |r_{ij}|} - 1 \right)
\]

where $r_{ij} = (WW_0^{-1})_{ij}$, $W_0$ is the known truth. $d(W, W_0)$ is equal to zero if and only $W$ and $W_0$ are equivalence. Besides Amari metrics, we also used the signal-to-interference ratio (SIR) function in ICA package [43]) as the criterion, where larger SIR indicates better performance in ICA.

**B. Audio separation task**

Two kinds of sources were used in the audio separation experiment: 6 speech recordings were from male (MJ60_07, MA03_01, MJ57_03) and female (FC14_04, FC18_06, FD19_06) speakers in TSP dataset [48], and 18 different random sources were from the past researches [9], [23]. These 24 sources ($N = 15200$) were mixed by an invertible matrix to produce the high dimensional observed mixtures $x$, and such procedure was replicated 50 times.

As can be seen from the Table II, B-SP, PICA, EICA acquired the best three separation performances, and FNA2 was actually not converged. Although B-SP performed slightly better than PICA in this experiment, we will later illustrate the simple way to improve our methods’ separation performance and time efficiency.

**C. Natural scene images separation task**

We designed an images separation experiment in this subsection, where the three gray-scale images were chosen from the ICS [49] package. These images depict a forest road, cat and sheep, and they have been used in many ICA researches [49]. We vectorized them to arrive into a $130^2 \times 3$ data matrix and we fixed the mixing matrix as $A = \begin{bmatrix} 0.8 & 0.3 & -0.3; 0.2 & -0.8 & 0.7; 0.3 & 0.2 & 0.3 \end{bmatrix}$ (column-wise).

The results are recorded in Table III where PICA performed better than B-SP ($df = 3, M = 5$) both in Amari/SIR and elapsed time. To improve the performance of B-SP, we have to level up its separation performance and cut down its elapsed time at the same time. Following the instructions in Subsection II-C, we successfully found the appropriate tuning parameters ($df = 8, M = 3$) for B-SP in few attempts, and we show them at the bottom of Table III.

**Table I**

ICA methods used in the experiments.

| Methods     | Symbols | Parameters |
|-------------|---------|------------|
| FastICA(G0) | F-G0    | $G_0(u) = \frac{1}{2} u^2$ |
| FastICA(G1) | F-G1    | $G_1(u) = \ln \cosh(u)$ |
| ProDenICA   | PICA    | default    |
| FixNA2      | FNA2    | default    |
| BoostingICA (smooth spline) | B-SP | $L = 500$, $df = 3$, $M = 5$ |
| EFICA       | EICA    | default    |

**Table II**

The Amari metrics (multiplied by 100), SIR and CPU elapsed time for audio separation task (50 replications).

|                        | F-G0     | F-G1     | PICA     | B-SP  | FNA2   | EICA   |
|------------------------|----------|----------|----------|-------|--------|--------|
| Amari metrics          | 30.51    | 30.64    | 12.41    | 13.67 | 414.86 | 29.53  |
| SIR                    | 21.98    | 23.64    | 30.14    | 29.30 | 7.02   | 27.16  |
| Elapsed time (s)       | 1.73     | 1.35     | 17.54    | 20.51 | 381.15 | 0.71*  |

**Table III**

The Amari metrics (multiplied by 100), SIR and CPU elapsed time for image separation task.

| Methods     | Amari metrics | SIR | Elapsed time (s) |
|-------------|---------------|-----|------------------|
| F-G0        | 38.61         | 10.69| 0.24             |
| F-G1        | 54.88         | 9.04 | 0.17             |
| PICA        | 19.04         | 16.39| 1.95             |
| B-SP ($df = 3, M = 5$) | 24.45 | 14.55| 2.62             |
| FNA2        | 32.17         | 11.84| 0.44             |
| EICA        | 37.87         | 10.11| 0.24*            |
| B-SP ($df = 5, M = 3$) | 19.90 | 15.91| 1.65             |
| B-SP ($df = 8, M = 3$) | **18.73** | **16.56** | **1.66** |

Once the running time is not our key interest, it might be an appropriate way to improve the separation performance via simply increasing $M$. The corresponding robust experiments on ICS data are shown in Figure 2 and B-SP outperformed PICA when $M \geq 48$.

**Fig. 2.** Separation performances of B-SP on ICS images mixtures. The ordinate is the Amari metrics/SIR, and the abscissa represents boosting iterations $M$. The performance of PICA is shown in the dot line for comparison, and B-SP overwhelmed PICA when $M \geq 48$.

**IV. Conclusion**

In this paper, we introduce boosting to the nonparametric independent component analysis to alleviate the density mismatching between unknown sources and their estimations. The proposed algorithm is based on our earlier research [50], and it is a joint likelihood maximization between boosting NPMLE and fixed-point method. The proposed method BoostingICA is concise and efficient, a list of experiments have illustrated its competitive performance with other popular or recent ICA algorithms.

1 code available: yunjil2sp.github.io
