A Qualitative Analysis of Independent Component Analysis Based Algorithms for the Removal of Artifacts from Electroencephalography Signals

1Balaiah Paulchamy, 2Ilavennila and 1Jayabal Jaya
1Department of Electronics and Communication Engineering, Hindusthan Institute of Technology, Coimbatore, Tamil Nadu, India
2Department of Electrical and Electronics Engineering, PSG College of Technology, Coimbatore, Tamil Nadu, India

Abstract: Problem statement: Independent Component Analysis (ICA) based algorithms applied in the context to remove the artifacts from the EEG signals are evaluated with appropriate metrics and it compares and contrasts the performance of the different methods for such applications. The primary goal is to gain some insight into relative performance of the various methods. Approach: CA is a statistical and computational technique for revealing hidden factors that underlie sets of random signals. In the ICA model the data samples are assumed to be linear mixture of some unknown latent variables and the mixing system is also unknown. The latent variables are assumed to have a nongaussian distribution. These variables are the independent components of the observed data which can be found, up to some degree of accuracy, using different algorithms based on ICA techniques. Results: The algorithms based on ICA with different approaches to be considered are JADE, Fast ICA, infomax and extended infomax and these performances are measured in terms of Entropy, PSNR and Speed. The simulation results show that the performance of each algorithm is to be preferred over another in terms of speed and reliability. A framework for accommodating four ICA algorithms is developed and hence selects the best algorithm for the specific type of data. Conclusion: ICA plays a vital role in removing of artifacts in EEG signals. It maintains the similarity in their patterns when subject is performing the mental task. The traditional methods applied for remove artifacts can only compromise between eliminating artifacts and protecting useful signals so that the result is not very satisfying. ICA method can protect the useful signals as well as obviously weaken even entirely remove the artifacts in multichannel EEG signals, this characteristic of ICA is the key to get stable EEG patterns which can be used for mental task classification.

Key words: Independent Component Analysis (ICA), Second Order Blind Identification (SOBI), Peak Signal to Noise Ratio (PSNR), First Order Blind Identification (FOBI), Additive White Gaussian Noise (AWGN)

INTRODUCTION

Electroencephalography (EEG) is a medical imaging technique that reads the scalp electrical activity generated by the brain structures (Binjadhnan and Ahmad, 2010), Electrical impulses generated by nerve firings in the brain diffuse through the head and can be measured by electrodes placed on the scalp. The EEG gives a coarse view of neural activity and has been used to non-invasively study cognitive processes and the physiology of the brain. The greatest advantage of EEG when compared with other medical imaging techniques is its speed.

When the EEG signals measured by electrodes placed on the scalp and are always under the influences of artifacts (Rizon, 2010). Those artifacts include: line noise from the power line, eye blinks, eye movements, heartbeat, breathing and other muscle activities. Because of these artifacts contained in EEG, the pattern detection in EEG produced from normal mental states is a very difficult problem.

MATERIALS AND METHODS

Independent component analysis: ICA is an extension of PCA but it is more powerful that PCA in the field of signal analysis. In mid 90’s several new
ICA algorithms (Furukawa, 2010; Hillyard and Galambos, 1970) were introduced with impressive demonstrations on problems like separating different speech signals from a mixed signal. The applications of ICA include but not limited to the fields of biomedical, telecommunications, audio and video signal processing feature extraction, data mining and functional time series analysis. Generally, ICA technique can be regarded as a technique to separate signals from a mixture. There are several ICA algorithms in use. Some of these algorithms are Fast ICA, JADE and First Order Blind Identification (FOBI), Maximum Likelihood and Infomax, algorithms based on Kernel methods and algorithms using time structure like Second Order Blind Identification (SOBI).

ICA model: A simple mathematical representation of ICA model (Ali et al., 2010) is as follows Consider a simple linear model which consists of N sources of T samples i.e., \( S = \{s(1), s(2), \ldots, s(T)\} \). The symbol there represents time but it may represent some other parameter like space. M weighted mixtures of the sources are observed as X, where \( X = \{x_1, x_2, \ldots, x_M\} \). This can be represented as in Eq. 1 and 2:

\[
x = Ax + n
\]

Where:

\[
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_M \\
\end{bmatrix}
= \begin{bmatrix}
s_1 \\
s_2 \\
\vdots \\
s_N \\
\end{bmatrix}
+ \begin{bmatrix}
n_1 \\
n_2 \\
\vdots \\
n_M \\
\end{bmatrix}
\]

And n represents the Additive White Gaussian Noise (AWGN). It is assumed that there are at least as many observations as sources i.e., \( M \geq N \). The \( M \times N \) matrix \( A \) is represented as in Eq. 3:

\[
A = \begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1N} \\
a_{21} & a_{22} & \cdots & a_{2N} \\
\vdots & \vdots & \ddots & \vdots \\
a_{M1} & a_{M2} & \cdots & a_{MN} \\
\end{bmatrix}
\]

A is called the mixing matrix. The estimation of the matrix \( S \) with knowledge of \( X \) is the linear source separation problem. This is schematically shown in Fig. 1. A is the mixing matrix and B is the unmixing matrix.

The following assumptions ensure that the ICA model estimates the independent components meaningfully. Actually the first assumption is the only true requirement which ICA demands. The other assumptions ensure that the estimated independent components are unique.

The latent variables (or independent components) are statistically independent and the mixing is linear. There is no more than one Gaussian signal among. The latent variables and the latent variables have cumulative density function not much different from a logistic sigmoid.

The number of observed signals, \( m \), is greater than or equal to the number of latent variables, \( n \) (i.e., \( m \geq n \)). If \( n > m \), we come to a special category of Independent Component Analysis called ICA with over-complete bases. In such a case the mixed signals do not have enough information to separate the independent components. There have been attempts to solve this particular problem but no rigorous proofs exist as of yet. If \( m > n \) then there is redundancy in the mixed signals. The ICA model works ideally when \( n = m \).

The mixing matrix is of full column rank, which means that the rows of the mixing matrix are linearly independent. If the mixing matrix is not of full rank then the mixed signals will be linear multiples of one another.

The propagation delay of the mixing medium is negligible. Before applying an ICA algorithm on the data, it is usually very useful to do some pre-processing (Ahmad and Ken, 2010).

Some pre-processing techniques that make the problem of ICA estimation simpler and better conditioned are given.

ICA algorithms-algebraic ICA algorithm: An algebraic solution to ICA is proposed by Taro Yamaguchi. This is a non-iterative algorithm but becomes extremely complex to compute when the number of sources goes more than two. For two sources separation it works very fast (Cichoki and Vorobyov, 2000). Two observed signals \( x_1 \) and \( x_2 \) are given by linear mixture of two independent original signals \( S_1 \) and \( S_2 \) as in Eq. 4:

\[
x_1 = a_{11}S_1 + a_{12}S_2
\]

\[
x_2 = a_{21}S_1 + a_{22}S_2
\]
\[
\begin{bmatrix}
X_1 \\
X_2
\end{bmatrix} = \begin{bmatrix}
10^{-s_1} \\
\beta l^{-s_2}
\end{bmatrix}
\]  
\hspace{1cm} (4)

where, \(\alpha\) and \(\beta\) are unknown mixing rates. The algebraic solution to \(\alpha\) and \(\beta\) are given by Eq. 5 and 6:

\[
\beta = \frac{\alpha C_2 - C_1}{\alpha C_1 - C_1}
\]  
\hspace{1cm} (5)

\[
(C_1 C_{10} - C_1 C_1)\alpha^2 + (3C_2 C_1 - 3C_2 C_2 - C_2 C_{10} + C_1 C_{11}) \alpha^3 + (3C_2 C_1 - 3C_2 C_2 - 3C_1 C_2) \alpha^2 + (C_1 C_1 - C_1 C_1) \alpha + (C_1 C_2 - C_2 C_1) = 0
\]  
\hspace{1cm} (6)

where, \(C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}\) and \(C_{11}\) are as shown in Eq. 7:

\[C_1 = E[X_1^2] - [E[X_1]]^2 = C_2[X_1] - [E[X_1]]^2\]
\[C_2 = E[X_1 X_2] - E[X_1] E[X_2]\]
\[C_3 = E[X_1 X_3] - E[X_1] E[X_3]\]
\[C_4 = E[X_2 X_2] - E[X_2^2] E[X_2]\]
\[C_5 = E[X_2 X_3] - E[X_2 X_3] E[X_3]\]
\[C_6 = E[X_3 X_3] - E[X_3^2] E[X_3]\]
\[C_7 = E[X_3 X_3] - E[X_3 X_3] E[X_3]\]
\[C_8 = E[X_1 X_4] - E[X_1] E[X_4]\]
\[C_9 = E[X_2 X_4] - E[X_2] E[X_4]\]
\[C_{10} = E[X_3 X_4] - E[X_3] E[X_4]\]
\[C_{11} = E[X_4 X_4] - E[X_4] E[X_4]\]

\hspace{1cm} (7)

where, \(E[\cdot]\) denotes the expectation operation. \(\alpha\) and \(\beta\) are obtained by solving the Eq. 5-7 with the Ferrari method. Excluding the solutions having non-zero imaginary parts and negative sizes the proper solution is selected (Hyvarinen, 1999). Original independent signals are computed from Eq. 6 by solving value of \(\alpha\) and \(\beta\).

**Fast ICA:** The advantage of using the gradient method to maximize negentropy is that the inputs \(z(t)\) can be used in the algorithm at once, thus enabling fast adaptation in non-stationary environment (Cichoki and Vorobyov, 2000). However convergence is slow and depends on a good choice of learning rate \(\gamma\).

To make this method efficient, a fast-fixed point algorithm is devised, also called Fast ICA for Negentropy. To understand this algorithm it should be noted that at a stable point of the gradient algorithm, the gradient must be pointing towards \(w\) or in other words it must be a scalar multiple of \(w\). That means that adding the gradient of negentropy in \(w\) would not change its direction at the stable points and hence convergence can be obtained. The Eq. 8 can be written as:

\[
W \leftarrow E[\mathbf{Z}^T W G^T W] W \leftarrow \frac{W}{E[W^2]}
\]  
\hspace{1cm} (8)

The coefficient \(\gamma\) is omitted because it would be eliminated by the normalization anyways. Iteration in Eq. 8 does not however have as good of a convergence as the one using kurtosis. The reason behind this is the non-polynomial moments (\(G'\)s) do not have same nice algebraic properties as cumulates like kurtosis. Hence to get a better convergent algorithm the iteration in Eq. 9 has to be modified. This modification can simply be done by adding some multiple of \(w\) to the both sides of the iterant term in Eq. 9 and then changing the value of multiple to find a better convergence speed as in Eq. 9:

\[
W = E[\mathbf{Z}^T G^T W z] \iff (1 + \alpha) W = E[\mathbf{Z}^T G^T W z] + \alpha E\]
\hspace{1cm} (9)

Adding a multiple of \(w\) to the both sides of Eq. 10 would not change the direction of the vector and after normalization in the next step \(w\) will be constrained to the unit sphere again. A suitable value of \(\alpha\) and thus the Fast ICA algorithm can be found using Newton’s method for solving equations (Berg and Scherg, 1991) Newton’s method can briefly explained as follows: To find a maxima or minima of any function with respect to some variable, first the function is expanded using Taylor’s series and the terms above the quadratic terms are dropped to keep it manageable (since higher order terms don’t contribute a lot in the total value of the function (Hyvarinen and Oja, 2000). Let \(E\) (not expectation) be a cost or error function which has to be minimized around vector \(\mathbf{w}\) having \(m\) elements and \(n\) being the number of iteration. The change in the cost function can be written as in Eq. 10:

\[
\Delta E(W^n) = E(W(n + 1) - E(W(n)) = E(W(n) + g^T(n))
\]
\[
\Delta W(n) + \frac{1}{2} \Delta W(n)^T H(n) \Delta W(n) - E(W(n))
\]  
\hspace{1cm} (10)

where, \(g(n) = m \times 1\) gradient vector of cost function evaluated at \(w(n)\) and, \(H(n)\) is an \(m \times m\) 2nd order derivative matrix of the cost function \(E(w(n))\) evaluated at \(w(n)\), called Hessian Matrix. Hessian Matrix \(H\) is given by Eq. 11:

\[
H = \Delta^2 E(W(n))
\]

where, \(g(n) = m \times 1\) gradient vector of cost function evaluated at \(w(n)\) and, \(H(n)\) is an \(m \times m\) 2nd order derivative matrix of the cost function evaluated at \(w(n)\) and, \(H(n)\) is an \(m \times m\) 2nd order derivative matrix of the cost function.
derivative matrix of the cost function \( E(w(n)) \) evaluated at \( w(n) \), called Hessian Matrix. Hessian Matrix \( H \) is given by Eq. 11:

\[
H = \Delta^2 E(W(n))
\]

Differentiating \( \Delta E(W(n)) \) w.r.t \( \Delta w(n) \) to find out the minimal value of \( \Delta E \) gives the condition as in Eq. 12:

\[
g(n) + H(n)\Delta W(n) = 0 \\
\Rightarrow \Delta W(n) = -H^{-1}(n)g(n) \\
W(n+1) = W(n) - H^{-1}(n)g(n)
\]

Expression in Eq. 12 is the Newton’s method for updating the vector \( w \) to move towards the minimization of the cost function. The advantage of Newton’s method is fast convergence but as it can be seen that it is computationally more intensive since one has to calculate inverse of Hessian matrix at each step.

In order to avoid the cost and time consuming calculation of the inverse of Hessian matrix in the Newton’s method, an approximation of this method is developed that avoids the use of matrix inversion without sacrificing its essence to employ ICA algorithm (Comon, 1994). The approximation of Newton’s method calls for the use of Lagrangian rule for constrained optimization. Lagrangian rule for constrained optimization can be briefly described as follows. Assume a cost function \( E(w) \) (E is not expectation) which is supposed to be minimized or maximized under some constraint \( H_i(w) = 0 \), where \( i = 1,2,3, \ldots, k \). One can write the Lagrangian function based on the given information as in Eq. 13:

\[
L(W,\lambda_1,\lambda_2,\ldots,\lambda_k) = E(W) + \sum_{i=1}^{k} \lambda_i H_i(W)
\]

where, \( \lambda_1,\lambda_2,\ldots,\lambda_k \) are called Lagrangian multipliers. The minimum (maximum) point of Eq. 14 where its gradient is zero with respect to both \( w \) and all of the \( \lambda_i \) gives the solution to the original constrained problem, i.e., minimization of \( E(w) \) under some constraint \( H_i(w) = 0 \). The gradient of \( L(W,\lambda_1,\lambda_2,\ldots,\lambda_k) \) with respect to \( \lambda_i \) gives the \( i^{th} \) constraint function \( H_i(w) \), so putting all these to zero will give the original constraint condition. When gradient of \( L(W,\lambda_1,\lambda_2,\ldots,\lambda_k) \) is taken with respect to \( w \) and equate it to zero, one will get the Eq. 14:

\[
\frac{\partial L}{\partial w}(W,\lambda_1,\lambda_2,\ldots,\lambda_k) = \frac{\partial E(W)}{\partial w} + \sum_{i=1}^{k} \lambda_i \frac{\partial H_i(W)}{\partial w} = 0
\]

Hence the minimization problem has been changed into two sets of equations that are much easier to solve. A possible way to solve these two sets, one given by the constraints, the other by Eq. 15, is some appropriate iteration method like Newton iteration. From Eq. 16 the optima of \( E(G(w^Tz)) \) for constraint \( ||w||^2 = 1 \) can be evaluated as in Eq. 15:

\[
\frac{\partial E[G(W^Tz)]}{\partial w} + \lambda \frac{||w||^2 - 1}{\partial w} = 0 \\
\Rightarrow E[zG(W^Tz)] + 2\lambda w = 0
\]

The second derivative of Lagrangian function can be evaluated as in Eq. 16:

\[
\frac{\partial^2 F}{\partial w^2} - E[zG^T(W^Tz)] + \beta I
\]

Thus the Newton iteration from Eq. 13 can be written as in Eq. 17:

\[
W \leftarrow W - \frac{\frac{\partial L}{\partial W}}{\frac{\partial F}{\partial W}} = W - \frac{E[zG(W^Tz)] + \beta W}{E[zG^T(W^Tz)] + \beta I}
\]

To simplify the calculations, since the vector \( z \) is shered then \( \frac{\partial F}{\partial w} \) can be approximated as in Eq. 18:

\[
\frac{\partial F}{\partial W} = E[zG^T(W^Tz)] + \beta I = E[zG^T(W^Tz)] + \beta I
\]

Hence the gradient becomes a diagonal matrix and can easily be inverted. Thus the algorithm becomes as in Eq. 19:
Multiplying both sides of Eq. 21 by $\text{E}[G(W^T z)] + \beta I$ and simplifying the resulting expression can be written as Eq. 20:

$$W[E(G(W^T z)) + \beta I] \leftarrow \text{E}[zG(W^T z)] - EG(W^T z)W$$  \hspace{1cm} (20)

Left hand side of Eq. 20 is nothing but a new variable to which right hand side value will be assigned, hence the Fast ICA algorithm based on negentropy will become as in Eq. 21:

$$W_{\text{new}} \leftarrow \text{E}[zG(W^T z)] - E[G(W^T z)] W_{\text{old}}$$  \hspace{1cm} (21)

Following is a brief summary of Fast ICA algorithm based on negentropy for finding one maximally non-Gaussian direction, i.e., estimating one independent component.

**JADE:** JADE is an algorithm that uses significant eigenpairs of the cumulant tensor $F(M)$ to find out the estimated values of independent components. In this algorithm the tensor eigenvalue decomposition is considered as more of a preprocessing step (Laubach et al., 1999). Eigenvalue decomposition can be viewed as diagonalization. The idea is to diagonalize $F(M)$ for any $M$ using the matrix $W$. In other words, $WF(M)W^T$ is diagonal. This is because the matrix $F$ consists of a linear combination of terms of the form $w_i w_i^T$, assuming that the ICA model holds. Hence the goal is to take a set of significant eigenmatrices, $M_i$ and try to make the matrices $WF(M)W^T$ as diagonal as possible. They might not be made exactly diagonal since the model doesn’t hold exactly because of some sampling errors. Let $Q = WF(M)W^T$, then maximization of the sum of the square of diagonal elements of the Eq. 22:

$$J_{\text{JADE}}(W) = \sum_i \| \text{diag}(WF(M_i))W^T \|^2$$  \hspace{1cm} (22)

Maximization of $J_{\text{JADE}}$ is then one method of joint approximate diagonalization of the $F(M_i)$.

Two of the main steps in the JADE algorithm are to find the significant eigenpairs of the cumulant tensor $\{ \lambda, r, M_r | 1 \leq r \leq n \}$ and to jointly diagonalize the JADE criterion $J_{\text{JADE}}(W)$. These two steps that lead to the JADE algorithm are discussed next.

**Infomax method:** The Infomax method has been proposed for performing linear ICA based on a principle of maximum information preservation (hence its name). However, it can also be seen as a maximum likelihood method, or as a method based on the minimization of mutual information. Infomax uses a network whose structure is depicted in Fig. 2 (the figure shows the case of two components; extension to a larger number of components is straightforward). $W$ is a linear block, yielding the Eq. 23:

$$Y = WX$$  \hspace{1cm} (23)

This block thus performs just a product by a square matrix (we shall designate both the block and the matrix by the same letter since this will cause no confusion). After optimization, the components of $Y$ are expected to be as independent from one another as possible. Blocks $\phi_i$ are auxiliary, being used only during the optimization phase (Lee et al., 2000). Each of them implements a nonlinear function (that we shall also designate by $\phi_i$). These functions must be increasing, with values in $[0; 1]$. The optimization of $W$ is made by maximizing the output entropy, $H(Z)$.

**Extended infomax method:** The algorithm of Bell and Sejnowski which uses a sigmoidal activation function is specifically suited to separate signals with super-Gaussian distribution (Alfaouri and Daqrouq, 2008). The proposed extension of Infomax ICA that is able to separate with sub and as well as super Gaussian distribution. This preserves the ICA architecture of Infomax algorithm, but it uses a learning rule derived by Girolami and Fyfe. It determines the sign changes (positive to negative and vice versa) required by the algorithm to handle both sub and super Gaussian distributions.

**RESULTS AND DISCUSSION**

Data were recorded for 10 sec during each task and each task was repeated five times per session (Verobyov and Cichocki, 2002; Vigon et al., 2000). Subjects attended two sessions recorded on separate weeks, resulting in a total of ten trials for each task. With a 250 Hz sampling rate, each 10 sec trial produces 2,500 samples per channel.
Fig. 3: EEG Recording of a Subject during Mental Multiplication

Figure 3 shows one Subject’s EEG Data obtained from doing maths tasks twice.

**Results of fast ICA algorithm:**

**Parameters:**

Nonlinearity: log (cosh(y))
No. of iterations: 100
Max. Weight change: 10e-300

Figure 4 shows the independent components obtained using the Fast ICA algorithm from the EEG data mixed with EOG which is shown in Fig. 3.

**Results of JADE algorithm:** No adjustable parameters. Execution time in seconds: Trail1: 0.99, Trail 2: 1.01. Figure 5 shows the independent components obtained using the JADE algorithm for the EEG data of Fig. 3.

**Results of infomax:**

Parameters: learning rate \(= 0.1\)
Max. Change in weight \(= 1e-3\)

Figure 6 shows the independent components obtained using the Infomax algorithm for the EEG data of Fig. 3.

**Results of Extended Informax:**

Parameters: Min. Weight-change: 1e-3
Number of iterations=512
Signs: -1: subgaussian
1: supergaussian
Initial weight= Identity matrix
Fig. 5: Independent components obtained using JADE (a) Trail-1 (b) Trail-2

Table 1: Comparative analysis of entropy between original EEG signal and ICA based Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Original</th>
<th>Fast ICA</th>
<th>JADE</th>
<th>Infomax</th>
<th>Ex-Infomax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trail 1</td>
<td>1.646</td>
<td>4.409</td>
<td>4.468</td>
<td>4.001</td>
<td>2.996</td>
</tr>
<tr>
<td>Trail 2</td>
<td>1.664</td>
<td>3.781</td>
<td>4.570</td>
<td>3.518</td>
<td>2.858</td>
</tr>
<tr>
<td>Trail 3</td>
<td>1.759</td>
<td>3.955</td>
<td>4.124</td>
<td>3.878</td>
<td>2.899</td>
</tr>
<tr>
<td>Trail 4</td>
<td>1.787</td>
<td>4.293</td>
<td>4.222</td>
<td>3.650</td>
<td>3.139</td>
</tr>
<tr>
<td>Trail 5</td>
<td>1.705</td>
<td>4.092</td>
<td>4.335</td>
<td>3.91</td>
<td>3.050</td>
</tr>
</tbody>
</table>

Nonlinearity = tanh(u)
Bias = 0
Learning rate = 0.1

Figure 7 shows the independent components obtained using the Extended Infomax algorithm for the EEG data of Fig. 3.

Performance comparisons of algorithms:

Computation time: The computation time i.e., the time taken by algorithm to separate the EEG signal is measured (Teplan, 2002). For comparison of algorithms mental multiplication task EEG (Kumar et al., 2008) which is measured for five trails is used. Entropy of original mental multiplication EEG signals of five trails is given in Table 1. Table 2 shows the comparative analysis between original EEG signal and ICA based algorithms. Table 3 shows computational time comparison in second.
CONCLUSION

ICA plays a vital role in removing of artifacts in EEG signals. It maintains the similarity in their patterns when subject is performing the mental task. BCI systems using EEG as control signal suffers from the artifact problem. The traditional methods applied for remove artifacts can only compromise between eliminating artifacts and protecting useful signals so that the result is not very satisfying. However, ICA method can protect the useful signals as well as obviously weaken even entirely remove the artifacts in multi-channel EEG signals, this characteristic of ICA is the key to get stable EEG patterns which can be used for mental task classification.

REFERENCES


