FASTER ICA UNDER ORTHOGONAL CONSTRAINT

Pierre Ablin[†]*, Jean-François Cardoso*[‡]*, and Alexandre Gramfort*[†]

[†] Inria, Parietal team, Université Paris-Saclay, Saclay, France
[‡] Institut d'Astrophysique de Paris, CNRS, Paris

ABSTRACT

Independent Component Analysis (ICA) is a technique for unsupervised exploration of multi-channel data widely used in observational sciences. In its classical form, ICA relies on modeling the data as a linear mixture of non-Gaussian independent sources. The problem can be seen as a likelihood maximization problem. We introduce Picard-O, a preconditioned L-BFGS strategy over the set of orthogonal matrices, which can quickly separate both super- and sub-Gaussian signals. It returns the same set of sources as the widely used FastICA algorithm. Through numerical experiments, we show that our method is faster and more robust than FastICA on real data.

Index Terms— Independent component analysis, blind source separation, quasi-Newton methods, maximum likelihood estimation, preconditioning.

1. INTRODUCTION

Independent component analysis (ICA) [1] is a popular technique for multi-sensor signal processing. Given an $N \times T$ data matrix X made of N signals of length T, an ICA algorithm finds an $N \times N$ 'unmixing matrix' W such that the rows of $Y = WX$ are 'as independent as possible'. An important class of ICA methods further constrains the rows of Y to be uncorrelated. Assuming zero-mean and unit variance signals, that is:

$$\frac{1}{T} Y Y^\top = I_N \ . \tag{1}$$

The 'whiteness constraint' (1) is satisfied if, for instance,

$$W = OW_0 \qquad W_0 = (\tfrac{1}{T} X X^\top)^{-1/2} \tag{2}$$

and the matrix O is constrained to be orthogonal: $OO^\top = I_N$. For this reason, ICA algorithms which enforce signal decorrelation by proceeding as suggested by (2) —whitening followed by rotation— can be called 'orthogonal methods'.

The orthogonal approach is followed by many ICA algorithms. Among them, FastICA [2] stands out by its simplicity, good scaling and a built-in ability to deal with both sub-Gaussian and super-Gaussian components. It also enjoys an impressive convergence speed when applied to data which are actual mixture of independent components [3, 4]. However, real data can rarely be accurately modeled as mixtures of independent components. In that case, the convergence of FastICA may be impaired or even not happen at all.

In [5], we introduced the Picard algorithm for fast non-orthogonal ICA on real data. In this paper, we extend it to an orthogonal version, dubbed Picard-O, which solves the same problem as FastICA, but faster on real data.

In section 2, the non-Gaussian likelihood for ICA is studied in the orthogonal case yielding the Picard Orthogonal (Picard-O) algorithm in section 3. Section 4 connects our approach to FastICA: Picard-O converges toward the fixed points of FastICA, yet faster thanks to a better approximation of the Hessian matrix. This is illustrated through extensive experiments on four types of data in section 5.

2. LIKELIHOOD UNDER WHITENESS CONSTRAINT

Our approach is based on the classical non-Gaussian ICA likelihood [6]. The $N \times T$ data matrix X is modeled as $X = AS$ where the $N \times N$ mixing matrix A is invertible and where S has statistically independent rows: the 'source signals'. Further, each row i is modeled as an i.i.d. signal with $p_i(\cdot)$ the probability density function (pdf) common to all samples. In the following, this assumption is denoted as the mixture model. It never perfectly holds for real problems. Under this assumption, the likelihood of A reads:

$$p(X|A) = \prod_{t=1}^{T} \frac{1}{|\det(A)|} \prod_{i=1}^{N} p_i([A^{-1}x]_i(t)) \ . \tag{3}$$

It is convenient to work with the negative averaged log-likelihood parametrized by the unmixing matrix $W = A^{-1}$, that is, $\mathcal{L}(W) = -\frac{1}{T} \log p(X|W^{-1})$. Then, (3) becomes:

$$\mathcal{L}(W) = -\log|\det(W)| - \hat{E}\left[\sum_{i=1}^{N} \log(p_i(y_i(t))\right] \ , \tag{4}$$

where $\hat{E}$ denotes the empirical mean (sample average) and where, implicitly, $[y_1, \cdots, y_N]^\top = Y = WX$.

We consider maximum likelihood estimation under the whiteness constraint (1). By (2) and (4), this is equivalent

This work was supported by the Center for Data Science, funded by the IDEX Paris-Saclay, ANR-11-IDEX-0003-02, and the European Research Council (ERC SLAB-YStG-676943).

to minimizing $\mathcal{L}(OW_0)$ with respect to the orthogonal matrix O. To do so, we propose an iterative algorithm. A given iterate $W_k = O_k W_0$ is updated by replacing O_k by a more likely orthonormal matrix O_{k+1} in its neighborhood. Following classical results of differential geometry over the orthogonal group [7], we parameterize that neighborhood by expressing O_{k+1} as $O_{k+1} = e^{\mathcal{E}} O_k$ where $\mathcal{E}$ is a (small) $N \times N$ skew-symmetric matrix: $\mathcal{E}^\top = -\mathcal{E}$.

The second-order Taylor expansion of $\mathcal{L}(e^{\mathcal{E}} W)$ reads:

$$\mathcal{L}(e^{\mathcal{E}} W) = \mathcal{L}(W) + \langle G|\mathcal{E}\rangle + \frac{1}{2}\langle \mathcal{E}|H|\mathcal{E}\rangle + \mathcal{O}(||\mathcal{E}||^3). \quad (5)$$

The first order term is controlled by the $N \times N$ matrix $G = G(Y)$, called relative gradient and the second-order term depends on the $N \times N \times N \times N$ tensor $H(Y)$, the relative Hessian. Both quantities only depend on $Y = WX$ and have simple expressions in the case of a small relative perturbation of the form $W \leftarrow (I + \mathcal{E})W$ (see [5] for instance). Those expressions are readily adapted to the case of interest, using the second order expansion $\exp(\mathcal{E}) \approx I + \mathcal{E} + \frac{1}{2}\mathcal{E}^2$. One gets:

$$G_{ij} = \hat{E}[\psi_i(y_i) y_j] - \delta_{ij} \ , \quad (6)$$

$$H_{ijkl} = \delta_{il}\delta_{jk} \hat{E}[\psi_i(y_i) y_i] + \delta_{ik}\, \hat{E}[\psi_i'(y_i) y_j y_l] \ . \quad (7)$$

where $\psi_i = -\frac{p_i'}{p_i}$ is called the *score function*.

This Hessian (7) is quite costly to compute, but a simple approximation is obtained using the form that it would take for large T and independent signals. In that limit, one has:

$$\hat{E}[\psi_i'(y_i) y_j y_l] \approx \delta_{jl}\, \hat{E}[\psi_i'(y_i)] \hat{E}[y_j^2] \quad \text{for} \quad i \neq j, \quad (8)$$

hence the approximate Hessian (recall $\hat{E}[y_j^2] = 1$):

$$\tilde{H}_{ijkl} = \delta_{il}\delta_{jk} \hat{E}[\psi_i(y_i) y_i] + \delta_{ik}\delta_{jl}\, \hat{E}[\psi_i'(y_i)] \ \text{ if } i \neq j. \quad (9)$$

Plugging this expression in the 2nd-order expansion (5) and expressing the result as a function of the $N(N-1)/2$ free parameters $\{\mathcal{E}_{ij},\ 1 \leq i < j \leq N\}$ of a skew-symmetric matrix $\mathcal{E}$ yields after simple calculations:

$$\langle G|\mathcal{E}\rangle + \frac{1}{2}\langle \mathcal{E}|\tilde{H}|\mathcal{E}\rangle = \sum_{i<j} (G_{ij} - G_{ji})\, \mathcal{E}_{ij} + \frac{\hat{\kappa}_i + \hat{\kappa}_j}{2}\, \mathcal{E}_{ij}^2 \quad (10)$$

where we have defined the non-linear moments:

$$\hat{\kappa}_i = \hat{E}[\psi_i(y_i) y_i] - \hat{E}[\psi_i'(y_i)] \ . \quad (11)$$

If $\hat{\kappa}_i + \hat{\kappa}_j > 0$ (this assumption will be enforced in the next section), the form (10) is minimized for $\mathcal{E}_{ij} = -(G_{ij} - G_{ji})/(\hat{\kappa}_i + \hat{\kappa}_j)$: the resulting quasi-Newton step would be

$$W_{k+1} = e^D W_k \quad \text{for} \quad D_{ij} = -\frac{2}{\hat{\kappa}_i + \hat{\kappa}_j} \frac{G_{ij} - G_{ji}}{2}. \quad (12)$$

That observation forms the keystone of the new orthogonal algorithm described in the next section.

3. THE PICARD-O ALGORITHM

As we shall see in Sec. 4, update (12) is essentially the behavior of FastICA near convergence. Hence, one can improve on FastICA by using a more accurate Hessian approximation. Using the exact form (7) would be quite costly for large data sets. Instead, following the same strategy as [5], we base our algorithm on the L-BFGS method (which learns the actual curvature of the problem from the data themselves) using approximation (10) only as a *pre-conditioner*.

L-BFGS and its pre-conditioning: The L-BFGS method keeps track of the m previous values of the (skew-symmetric) relative moves $\mathcal{E}_k$ and gradient differences $\Delta_k = (G_k - G_k^\top)/2 - (G_{k-1} - G_{k-1}^\top)/2$ and of auxiliary quantities $\rho_k = \langle \mathcal{E}_k | \Delta_k \rangle$. It returns a descent direction by running through one backward loop and one forward loop. It can be preconditioned by inserting a Hessian approximation in between the two loops as summarized in algorithm 1.

Stability: If the ICA mixture model holds, the sources should constitute a local minimum of $\mathcal{L}$. According to (10), that happens if $\hat{\kappa}_i + \hat{\kappa}_j > 0$ for all $i < j$ (see also [8]). We enforce that property by taking, at each iteration:

$$\psi_i(\cdot) = \text{sign}(k_i)\psi(\cdot) \quad (13)$$

where $k_i = \hat{E}[\psi(y_i) y_i] - \hat{E}[\psi'(y_i)]$ and ψ is a fixed nonlinearity (a typical choice is $\psi(u) = \tanh(u)$). This is very similar to the technique in extended Infomax [9]. It enforces $\hat{\kappa}_i = |k_i| > 0$, and the positive definiteness of $\tilde{H}$. In practice, if for any signal i the sign of k_i changes from one iteration to the next, L-BFGS's memory is flushed.

Regularization: The switching technique guarantees that the Hessian approximation is positive, but one may be wary of very small values of $\hat{\kappa}_i + \hat{\kappa}_j$. Hence, the pre-conditioner uses a floor: $\max((\hat{\kappa}_i + \hat{\kappa}_j)/2, \kappa_{\min})$ for some small positive value of $\kappa_{\min}$ (typically $\kappa_{\min} \simeq 10^{-2}$).

Line search: A backtracking line search helps convergence. Using the search direction D returned by L-BFGS, and starting from a step size $\alpha = 1$, if $\mathcal{L}(\exp(\alpha D)W) < \mathcal{L}(W)$, then set $\mathcal{E} = \alpha D$ and $W \leftarrow \exp(\mathcal{E})W$, otherwise divide α by 2 and repeat.

Stopping: The stopping criterion is $||G - G^\top|| < \varepsilon$ where ε is a small tolerance constant.

Combining these ideas, we obtain the Preconditioned Independent Component Analysis for Real Data-Orthogonal (Picard-O) algorithm, summarized in table 2.

The Python code for Picard-O is available online at `https://github.com/pierreablin/picard`.

4. LINK WITH FASTICA

This section briefly examines the connections between Picard-O and symmetric FastICA [10]. In particular, we show that

Algorithm 1: Two-loop recursion L-BFGS formula

Input : Current gradient G_k, moments $\hat{\kappa}_i$, previous $\mathcal{E}_l$, Δ_l, ρ_l $\forall l \in \{k-m, \dots, k-1\}$.

Set $Q = -(G_k - G_k^\top)/2$;

for *l=k-1,…,k-m* **do**

  Compute $a_l = \rho_l \langle \mathcal{E}_l | Q \rangle$;

  Set $Q = Q - a_l \Delta_i$;

end

Compute D as $D_{ij} = Q_{ij} / \max\left(\frac{\hat{\kappa}_i \hat{\kappa}_j}{2}, \kappa_{\min}\right)$;

for *l=k-m,…,k-1* **do**

  Compute $\beta = \rho_l \langle \Delta_l | D \rangle$;

  Set $D = D + \mathcal{E}_l(a_l - \beta)$;

end

Output: Descent direction D

Algorithm 2: The Picard-O algorithm

Input : Initial signals X, number of iterations K

Sphering: compute W_0 by (1) and set $Y = W_0 X$;

for $k = 0 \cdots K$ **do**

  Compute the signs $\operatorname{sign}(k_i)$;

  Flush the memory if the sign of any source has changed ;

  Compute the gradient G_k ;

  Compute search direction D_k using algorithm 1 ;

  Compute the step size α_k by line search ;

  Set $W_{k+1} = \exp(\alpha_k D_k) W_k$ and $Y = W_{k+1} X$;

  Update the memory;

end

Output: Unmixed signals Y, unmixing matrix W_k

both methods essentially share the same solutions and that the behavior of FastICA is similar to a quasi-Newton method.

Recall that FastICA is based on an $N \times N$ matrix $C(Y)$ defined entry-wise by:

$$C_{ij}(Y) = \hat{E}[\psi_i(y_i) y_j] - \delta_{ij} \hat{E}[\psi_i'(y_i)] \ . \tag{14}$$

The symmetric FastICA algorithm, starting from white signals Y, can be seen as iterating $Y \leftarrow C_w(Y) Y$ until convergence, where $C_w(Y)$ is the orthogonal matrix computed as

$$C_w(Y) = (CC^\top)^{-\frac{1}{2}} C \ . \tag{15}$$

In the case of a fixed score function, a sign-flipping phenomenon appears leading to the following definition of a fixed point: Y is a fixed point of FastICA if $C_w(Y)$ is a diagonal matrix of ± 1 [11]. This behavior can be fixed by changing the score functions as in (13). It is not hard to see that, if ψ is an odd function, such a modified version has the same trajectories as the fixed score version (up to to irrelevant sign flips), and that the fixed points of the original algorithm now all verify $C_w(Y) = I_N$.

Stationary points : We first relate the fixed points of FastICA (or rather the sign-adjusted version described above) to the stationary points of Picard-O.

Denote C_+ (resp. C_-) the symmetric (resp. skew-symmetric) part of $C(Y)$ and similarly for G. It follows from (6) and (14) that

$$C = C_+ + C_- = C_+ + G_-$$

since $C_- = G_- = (G - G^\top)/2$.

One can show that Y is a fixed point of FastICA if and only if $G(Y)$ is symmetric and $C_+(Y)$ is positive definite. Indeed, at a fixed point, $C_w(Y) = I_N$, so that by Eq. (15), one has $C(Y) = (CC^\top)^{1/2}$ which is a positive matrix (almost surely). Conversely, if $G(Y)$ is symmetric, then so is $C(Y)$. If $C(Y)$ is also positive, then its polar factor $C_w(Y)$ is the identity matrix, so that Y is a fixed point of FastICA.

The modification of FastICA ensures that the diagonal of $C(Y)$ is positive, but does not guarantee positive definiteness. However, we empirically observed that on each dataset used in the experiments, the matrix $C_+(Y)$ is positive definite when $G_-(Y)$ is small. Under that condition, we see that the stationary points of Picard-O, characterized by $G_-(Y) = 0$ are exactly the fixed points of FastICA.

Asymptotic behavior of FastICA : Let us now expose the behavior of FastICA close to a fixed point *i.e.* when $C_w(Y) = \exp(\mathcal{E})$ for some small skew-symmetric matrix $\mathcal{E}$.

At first order in $\mathcal{E}$, the polar factor $C_w = \exp(\mathcal{E})$ of C is obtained as solution of (proof omitted):

$$G_- = \frac{C_+ \mathcal{E} + \mathcal{E} C_+}{2} \ . \tag{16}$$

Denote by $\hat{H}$ the linear mapping $\hat{H} : \mathcal{E} \to -\frac{C_+ \mathcal{E} + \mathcal{E} C_+}{2}$. When FastICA perform a small move, it is (at first order) of the form $W \leftarrow e^D W$ with $D = -\hat{H}^{-1}(G_-)$. It corresponds to a quasi-Newton step with $\hat{H}$ as approximate Hessian.

Furthermore, under the mixture model assumption, close from separation and with a large number of samples, C_+ becomes the diagonal matrix of coefficients $\delta_{ij} \hat{\kappa}_i$ and $\hat{H}$ simplifies, giving the same direction D given in (12).

In summary, we have shown that a slightly modified version of FastICA (with essentially the same iterates as the original algorithm) has the same fixed points as Picard-O. Close to such a fixed point, each of FastICA's iteration is similar to a quasi-Newton step with an approximate Hessian. This approximation matches the true Hessian if the mixture model holds, but this cannot be expected in practice on real data.

5. EXPERIMENTS

This section illustrates the relative speeds of FastICA and Picard-O. Both algorithms are coded in the Python programming language. Their computation time being dominated by score evaluations, dot products and sample averages, a fair

comparison is obtained by ensuring that both algorithms call the exact same procedures so that speeds differences mostly stems from algorithmics rather than implementation.

Figure 1 summarizes our results. It shows the evolution of the projected gradient norm $||G - G^\top||$ versus iterations (left column) and versus time (right column). The 4 rows correspond to 4 data types: synthetic mixtures, fMRI signals, EEG signals, and image patches. FastICA and Picard-O speeds are compared using $\psi(\cdot) = \tanh(\cdot)$. The signals are centered and whitened before running ICA.

Experiments are repeated several times for each setup. The solid line shows the median of the gradient curves and the shaded area shows the $10\ \% - 90\ \%$ percentile (meaning that half the runs completed faster than the solid line and that 80% have convergence curves in the shaded area).

Synthetic data We generate $N = 50$ i.i.d. sources of length $T = 10000$. The 25 first sources follow a uniform law between -1 and 1, the 25 last follow a Laplace law ($p \propto \exp(-|x|)$). The $N \times T$ source matrix S is multiplied by a random square mixing matrix A. This experiment is repeated 100 times, changing each time the seed generating the signals and the mixing matrix.

fMRI This is functional MRI data processed by group ICA [12]. The datasets come from ADHD-200 consortium [13]. The problem is of size $N = 60$, $T = 60000$, and the experiments are repeated over 20 datasets.

EEG ICA is applied on 13 publicly available[1] electroencephalography datasets [14]. Each recording contains $N = 71$ signals, of length $T \simeq 75000$.

Image patches We use a database of 80 different images of open country [15]. From each image, $T = 10000$ patches of size 8×8 are extracted and vectorized to obtain $N = 64$ signals, before applying ICA.

Results. FastICA is slightly faster than Picard-O on the simulated problem, for which the ICA mixture model holds perfectly. However, on real data, the rate of convergence of FastICA is severely impaired because the underlying Hessian approximation is far from the truth, while our algorithm still converges quickly. Picard-O is also more consistent in its convergence pattern, showing less spread than FastICA.

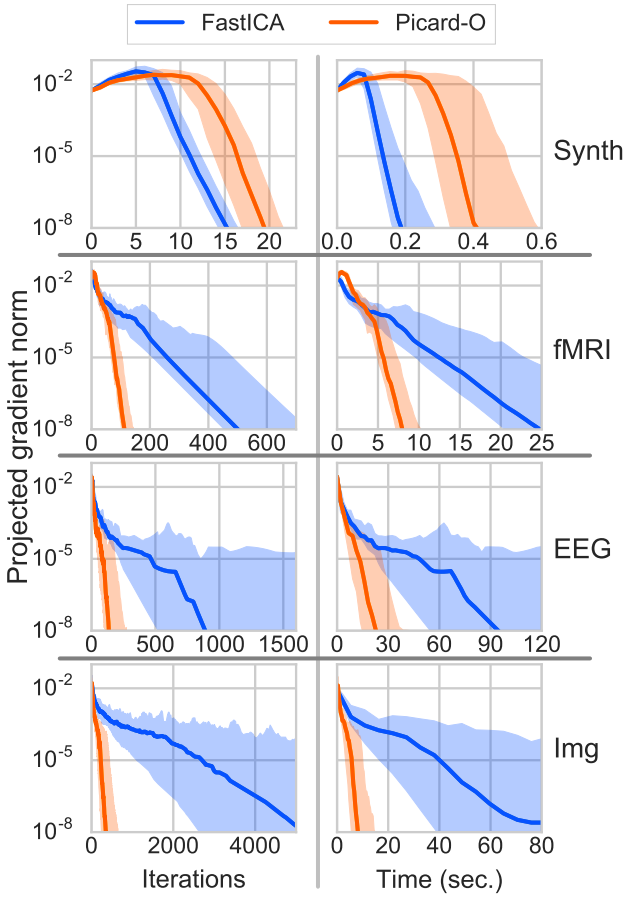

Fig. 1. Comparison between FastICA and Picard-O. Gradient norm vs iterations (left column) and vs time (right column). From top to bottom: simulated data, fMRI data, EEG data and image data. Solid line corresponds to the median of the runs, the shaded area covers the $10\% - 90\%$ percentiles.

6. DISCUSSION

In this paper, we show that, close from its fixed points, FastICA's iterations are similar to quasi-Newton steps for maximizing a likelihood. Furthermore, the underlying Hessian approximation matches the true Hessian of the problem if the signals are independent. However, on real datasets, the independence assumption never perfectly holds. Consequently, FastICA may converge very slowly on applied problems [16] or can get stuck in saddle points [17]. In [18], an exact line-search technique is proposed to accelerate FastICA, but it only works with a specific score function $\psi(x) = x^3$, and does not provide better descent directions.

We propose the Picard-O algorithm: as an extension of [5], it uses a preconditioned L-BFGS technique to build a good Hessian approximation and solve the same minimization problem as FastICA. Extensive experiments on three types of real data demonstrate that Picard-O can be orders of magnitude faster than FastICA.

[1] https://sccn.ucsd.edu/wiki/BSSComparison

7. REFERENCES

[1] P. Comon, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287 – 314, 1994.

[2] A. Hyvarinen, "Fast and robust fixed-point algorithms for independent component analysis," *IEEE Transactions on Neural Networks*, vol. 10, no. 3, pp. 626–634, 1999.

[3] E. Oja and Z. Yuan, "The fastica algorithm revisited: Convergence analysis," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1370–1381, 2006.

[4] H. Shen, M. Kleinsteuber, and K. Huper, "Local convergence analysis of FastICA and related algorithms," *IEEE Transactions on Neural Networks*, vol. 19, no. 6, pp. 1022–1032, 2008.

[5] P. Ablin, J.-F. Cardoso, and A. Gramfort, "Faster independent component analysis by preconditioning with hessian approximations," *Arxiv Preprint*, 2017.

[6] D. T. 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, 1997.

[7] W. Bertram, "Differential Geometry, Lie Groups and Symmetric Spaces over General Base Fields and Rings," *Memoirs of the American Mathematical Society*, 2008.

[8] J.-F. Cardoso, "On the stability of some source separation algorithms," in *Proc. of the 1998 IEEE SP workshop on neural networks for signal processing (NNSP '98)*, 1998, pp. 13–22.

[9] T.-W. Lee, M. Girolami, and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources," *Neural computation*, vol. 11, no. 2, pp. 417–441, 1999.

[10] A. Hyvärinen, "The fixed-point algorithm and maximum likelihood estimation for independent component analysis," *Neural Processing Letters*, vol. 10, no. 1, pp. 1–5, 1999.

[11] T. Wei, "A convergence and asymptotic analysis of the generalized symmetric FastICA algorithm," *IEEE transactions on signal processing*, vol. 63, no. 24, pp. 6445–6458, 2015.

[12] G. Varoquaux, S. Sadaghiani, P. Pinel, A. Kleinschmidt, J.-B. Poline, and B. Thirion, "A group model for stable multi-subject ICA on fMRI datasets," *Neuroimage*, vol. 51, no. 1, pp. 288–299, 2010.

[13] A.-. Consortium, "The ADHD-200 consortium: a model to advance the translational potential of neuroimaging in clinical neuroscience," *Frontiers in systems neuroscience*, vol. 6, 2012.

[14] A. Delorme, J. Palmer, J. Onton, R. Oostenveld, and S. Makeig, "Independent EEG sources are dipolar," *PloS one*, vol. 7, no. 2, p. e30135, 2012.

[15] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International journal of computer vision*, vol. 42, no. 3, pp. 145–175, 2001.

[16] P. Chevalier, L. Albera, P. Comon, and A. Ferréol, "Comparative performance analysis of eight blind source separation methods on radiocommunications signals," in *Proc. of IEEE International Joint Conference on Neural Networks*, vol. 1, 2004, pp. 273–278.

[17] P. Tichavsky, Z. Koldovsky, and E. Oja, "Performance analysis of the FastICA algorithm and cramer-rao bounds for linear independent component analysis," *IEEE transactions on Signal Processing*, vol. 54, no. 4, pp. 1189–1203, 2006.

[18] V. Zarzoso and P. Comon, "Robust independent component analysis by iterative maximization of the kurtosis contrast with algebraic optimal step size," *IEEE Transactions on Neural Networks*, vol. 21, no. 2, pp. 248–261, 2010.