ICA Using Kernel Entropy Estimation with NlogN Complexity*

Sarit Shwartz, Michael Zibulevsky, and Yoav Y. Schechner

Department of Electrical Engineering
Technion - Israel Institute of Technology, Haifa 32000, Israel
psarit@tx.technion.ac.il, {mzib,yoav}@ee.technion.ac.il

Abstract. Mutual information (MI) is a common criterion in independent component analysis (ICA) optimization. MI is derived from probability density functions (PDF). There are scenarios in which assuming a parametric form for the PDF leads to poor performance. Therefore, the need arises for non-parametric PDF and MI estimation. Existing non-parametric algorithms suffer from high complexity, particularly in high dimensions. To counter this obstacle, we present an ICA algorithm based on accelerated kernel entropy estimation. It achieves both high separation performance and low computational complexity. For K sources with N samples, our ICA algorithm has an iteration complexity of at most $\mathcal{O}(KN \log N + K^2 N)$.

1 Introduction

Mutual information (MI) of signals is a natural criterion for statistical dependency and is thus used in ICA algorithms (see for example in [5, 2, 7, 9, 10] and references therein). MI is based on an estimate of the probability density function (PDF) of signals, which is computationally costly. For this reason, existing ICA algorithms have assumed rough models for the PDFs [1, 4, 8], or used high order cumulants instead of MI [3]. These approximations can sometimes lead to failure, as demonstrated in [2] as well as in our current paper. In contrast, rather robust separation can be achieved with non-parametric kernel-based estimation of PDFs [2]. The drawback of that algorithm is high computational complexity. For K sources, each of which having N samples, that algorithm has a complexity of $\mathcal{O}(K^2 N^2)$. Another existing algorithm [7] has a complexity of $\mathcal{O}(3^K N + K^2 N)$, which may be tolerated for a small K, but has exponential growth in K.

In this study, we develop non-parametric ICA that has $\mathcal{O}(KN \log N + K^2 N)$ complexity by using an approximation of the kernel estimator. The approxima-

* This research was supported by the HASSIP Research Network Program HPRN-CT-2002-00285, sponsored by the European Commission. It was also supported by the US-Israel Binational Science Foundation (BSF), and the Ollendorff Minerva Center. Minerva is funded through the BMBF. Yoav Schechner is a Landau Fellow - supported by the Taub Foundation, and an Alon Fellow.

C.G. Puntonet and A. Prieto (Eds.): ICA 2004, LNCS 3195, pp. 422–429, 2004.

tion is calculated using a fast convolution. The errors caused by the approximation are reasonably small. Therefore, our method makes non-parametric ICA a practical algorithm for large problems.

2 Blind Source Separation and Mutual Information

Let $\{\mathbf{s}_1, \mathbf{s}_2, ...\mathbf{s}_K\}$ be a set of independent sources. Each source is of the form $\mathbf{s}_k = [s_k(1), s_k(2), \ldots, s_k(N)]^T$. Let $\{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_K\}$ be a set of measured signals, each of which being a linear mixture of the sources. Denote $\{\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ...\hat{\mathbf{s}}_K\}$ as the set of the reconstructed sources and $\mathbf{W}$ as the separation matrix. Then,

$$[\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ..., \hat{\mathbf{s}}_K]^T = \mathbf{W}[\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_K]^T \ . \tag{1}$$

The MI of the K random variables $\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ...\hat{\mathbf{s}}_K$ is (see for example [5])

$$\mathcal{I}(\hat{\mathbf{s}}_1, \hat{\mathbf{s}}_2, ...\hat{\mathbf{s}}_K) = \mathcal{H}_{\hat{\mathbf{s}}_1} + \mathcal{H}_{\hat{\mathbf{s}}_2} + ... + \mathcal{H}_{\hat{\mathbf{s}}_K} - \log|\det(\mathbf{W})| - \mathcal{H}_{\text{measurements}} \ , \tag{2}$$

where $\mathcal{H}(\hat{\mathbf{s}}_k)$ is the differential entropy (DE) of $\hat{\mathbf{s}}_k$. Here, $\mathcal{H}_{\text{measurements}}$ is independent of $\mathbf{W}$ and is constant for a given sample set $\{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_K\}$. Thus, the minimization problem that we solve is

$$\min_{\mathbf{W}} \left\{ \sum_{k=1}^{K} \mathcal{H}_{\hat{\mathbf{s}}_k} - \log|\det(\mathbf{W})| + \lambda \sum_{k=1}^{K} (\|\hat{\mathbf{s}}_k\| - 1)^2 \right\} \ . \tag{3}$$

The last sum $\sum(\|\hat{\mathbf{s}}_k\| - 1)^2$ in Eq. (3) weighted by a constant λ penalizes for un-normalized sources, therefore resolving ambiguities arising from the scale invariance of MI[1]. The gradient of this normalization penalty term is trivial to calculate and efficient to implement [11]. Therefore we do not discuss it further.

For non-parametric estimation of the DEs $\mathcal{H}_{\hat{\mathbf{s}}_k}$, we use the Parzen-windows estimator [2, 12]. That estimator has a high computational complexity. Our method bypasses this problem using FFT-based fast convolution.

3 Estimation of MI and Its Gradient

Estimating DE using Parzen-windows [12] enables us to differentiate the estimated entropies, and have a closed form expression for the DE gradients. The Parzen-window estimator for the PDF at a value t is

$$\hat{p}(t|\hat{\mathbf{s}}_k) \equiv (1/N) \sum_{n=1}^{N} \varphi[t - \hat{s}_k(n)] \ , \tag{4}$$

where $\hat{s}_k(n)$ is a sample from $\hat{\mathbf{s}}_k$ and $\varphi(t)$ is a smoothing kernel[2]. The Parzen-windows estimator [2, 12] for the DE of $\hat{\mathbf{s}}_k$ is

$$\hat{\mathcal{H}}_{\hat{\mathbf{s}}_k} = -\frac{1}{N} \sum_{l=1}^{N} \log \hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k] \ , \tag{5}$$

[1] This term does not affect the separation quality, but improves convergence of the optimization algorithm [11].

[2] We use a Gaussian kernel with a zero mean and variance σ^2. Following [2], we use $\sigma = 1.06N^{-1/5}$.

explicitly,

$$\hat{\mathcal{H}}_{\hat{\mathbf{s}}_k} = -\frac{1}{N}\sum_{l=1}^{N}\log\left\{\frac{1}{N}\sum_{n=1}^{N}\varphi[\hat{s}_k(l)-\hat{s}_k(n)]\right\}\,, \tag{6}$$

The gradient of $\log|\det(\mathbf{W})|$ is $(\mathbf{W}^{-1})^T$ (see for example [5]). Therefore, the MI gradient is

$$\nabla_{\mathbf{W}}\mathcal{I}(\hat{\mathbf{s}}_1,\hat{\mathbf{s}}_2,...\hat{\mathbf{s}}_K) = \nabla_{\mathbf{W}}\sum_{k=1}^{K}\mathcal{H}_{\hat{\mathbf{s}}_k} - (\mathbf{W}^{-1})^T\,. \tag{7}$$

We calculate the gradients of the sum of DEs $\nabla_{\mathbf{W}}\sum_{k=1}^{K}\mathcal{H}_{\hat{\mathbf{s}}_k}$ in two stages, using a chain rule. First, we calculate the DEs gradients with respect to the estimated sources

$$\nabla_{\hat{\mathbf{s}}_k}\mathcal{H}_{\hat{\mathbf{s}}_k} = \left[\frac{\partial\mathcal{H}_{\hat{\mathbf{s}}_k}}{\partial\hat{s}_k(1)},\ldots,\frac{\partial\mathcal{H}_{\hat{\mathbf{s}}_k}}{\partial\hat{s}_k(N)}\right]^T\,. \tag{8}$$

Then, we calculate the gradients of the sum of DEs with respect to the separation matrix by

$$\nabla_{\mathbf{W}}\sum_{k=1}^{K}\mathcal{H}_{\hat{\mathbf{s}}_k} = [\nabla_{\hat{\mathbf{s}}_1}\mathcal{H}_{\hat{\mathbf{s}}_1},...,\nabla_{\hat{\mathbf{s}}_K}\mathcal{H}_{\hat{\mathbf{s}}_K}]^T[\mathbf{y}_1,...,\mathbf{y}_K]\,. \tag{9}$$

The derivatives of Eq. (6) are

$$\begin{aligned}\frac{\partial\mathcal{H}_{\hat{\mathbf{s}}_k}}{\partial\hat{s}_k(r)} &= -\frac{1}{N}\sum_{l=1}^{N}\frac{\frac{1}{N}\sum_{n=1}^{N}\varphi'[\hat{s}_k(l)-\hat{s}_k(n)][\delta_{lr}-\delta_{nr}]}{\frac{1}{N}\sum_{n=1}^{N}\varphi[\hat{s}_k(l)-\hat{s}_k(n)]} = \\ &= -\frac{1}{N}\frac{\frac{1}{N}\sum_{n=1}^{N}\varphi'[\hat{s}_k(r)-\hat{s}_k(n)]}{\hat{p}[\hat{s}_k(r)|\hat{\mathbf{s}}_k]} + \frac{1}{N}\sum_{l=1}^{N}\frac{\frac{1}{N}\varphi'[\hat{s}_k(l)-\hat{s}_k(r)]}{\hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k]}\,,\end{aligned} \tag{10}$$

where δ_{lr} is the Kroneker delta, φ' is the derivative of φ, and $\hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k]$ is defined in Eq. (4). Define

$$\Phi'[\hat{s}_k(l)|\hat{\mathbf{s}}_k] \equiv \frac{1}{N}\sum_{n=1}^{N}\varphi'[\hat{s}_k(l)-\hat{s}_k(n)]\,, \tag{11}$$

$$F'[\hat{s}_k(l)] \equiv \frac{1}{N}\sum_{n=1}^{N}\frac{\varphi'[\hat{s}_k(n)-\hat{s}_k(l)]}{\hat{p}[\hat{s}_k(n)|\hat{\mathbf{s}}_k]}\,. \tag{12}$$

Then, Eq. (10) can be written as

$$\frac{\partial\mathcal{H}_{\hat{\mathbf{s}}_k}}{\partial\hat{s}_k(l)} = -\frac{1}{N}\frac{\Phi'[\hat{s}_k(l)|\hat{\mathbf{s}}_k]}{\hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k]} + \frac{1}{N}F'[\hat{s}_k(l)]\,, \tag{13}$$

Calculating the MI gradient explicitly using Eqs. (7-13) has a complexity of $\mathcal{O}(KN^2+K^2N)$, for details see [11]. This complexity is achieved thanks to the exploiting of the chain rule (Eq. 9) and it is lower than the $\mathcal{O}(K^2N^2)$ complexity of gradient calculation presented in [2].

4 Efficient Calculation of the Entropy Estimator

The PDF estimator given by Eq. (4) can be seen as a convolution

$$\hat{p}(t|\hat{\mathbf{s}}_k) = f * \varphi \,, \tag{14}$$

where

$$f(t) = (1/N) \sum_{n=1}^{N} \delta[t - \hat{s}_k(n)] \,. \tag{15}$$

It requires N^2 calculations of φ to compute this convolution in N points as needed in Eq. (5). On the other hand, it is known that fast convolution can be performed in $\mathcal{O}(N \log N)$ operations if done over a *uniform grid*. Therefore we resample (interpolate) the function $f(t)$ to a uniform grid. Then we convolve it with a uniformly sampled version of φ, which we denote φ_{sampled}. Finally, we interpolate the results back to the set of points $\hat{s}_k(l)$ used in entropy calculation Eq. (5). This process is illustrated in Fig. 1. The resampling of f starts by defining a vote function $\mathbf{v}$ on a uniform grid of length M, with a step size of Δ_v. Let $m^{\#}$ be the index of the grid node closest to the value of $\hat{s}_k(n)$ that satisfies

$$m^{\#} \leq \hat{s}_k(n)/\Delta_v \leq m^{\#} + 1 \,. \tag{16}$$

Define the distance of $\hat{s}_k(n)$ from the index $m^{\#}$ (normalized by Δ_v) by

$$\eta = \frac{\hat{s}_k(n)}{\Delta_v} - m^{\#} \,, \quad 0 \leq \eta \leq 1 \ . \tag{17}$$

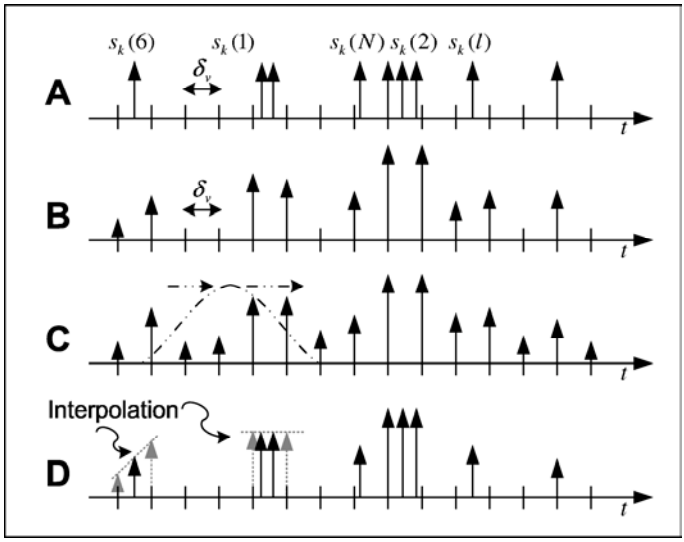

Fig. 1. Efficient calculation of $\hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k]$: (A) The function f. (B) The result of voting is a function on a uniform grid. (C) The result of discrete convolution with the sampled kernel. (D) Interpolation to the original $\hat{s}_k(l)$.

Let $h(\eta)$ be a function[3] that satisfies $h(1-\eta) = 1-h(\eta)$. Then, for each sample $\hat{s}_k(n)$ we update the vote function by

$$v(m) \leftarrow \begin{cases} v(m) + h(\eta) & \text{for } \; m = m^{\#} \\ v(m) + 1 - h(\eta) & \text{for } \; m = m^{\#} + 1 \end{cases} . \tag{18}$$

After the voting is over, we associate $\mathbf{v}/N$ with the resampled f. This transfers the function illustrated in Fig. 1A to the function illustrated in Fig. 1B. Then, we convolve[4] $\mathbf{v}/N$ with φ_{sampled} (Fig. 1B, $\rightarrow$ Fig. 1C).

$$\hat{p}_{\text{u}} = (\mathbf{v}/N) * \varphi_{\text{sampled}} . \tag{19}$$

Note that $\hat{p}_{\text{u}}$ resides on the uniform grid. However, the DE (Eq. 5) does not use $\hat{p}_{\text{u}}$, but rather $\hat{p}[\hat{s}_k(l)]$. We obtain an estimation of $\hat{p}[\hat{s}_k(l)]$ by interpolating the values of $\hat{p}_{\text{u}}$ onto the points $\hat{s}_k(l)$, using the same interpolation function $h(\eta)$ as before

$$\hat{p}[\hat{s}_k(l)|\hat{\mathbf{s}}_k] = h(\eta)\hat{p}_{\text{u}}(m^{\#}) + [1-h(\eta)]\hat{p}_{\text{u}}(m^{\#}+1) , \tag{20}$$

where $m^{\#}$ and η are defined in (16, 17). This step is illustrated in Fig. 1D. Finally, we estimate the DE by Eq. (5).

The voting, the interpolation and the entropy calculation (Eqs. 5,18 and 20) require $\mathcal{O}(N)$ operations. The convolution (Eq. 20) requires $\mathcal{O}(M \log N_{\text{kernel}})$ operations[5], where N_{kernel} is the length of φ_{sampled}. In addition, estimating the sources (Eq. 1) requires $\mathcal{O}(K^2 N)$ operations. Therefore, the overall complexity of calculating the DEs for the K estimated sources is $\mathcal{O}(KM \log N_{\text{kernel}} + K^2 N)$. This is significantly lower than the $\mathcal{O}(KN^2 + K^2 N)$ complexity of the explicit calculation of the K DEs using Eq. (6).

5 Efficient Estimation of the Entropy Gradient

Calculating the DE gradient explicitly using Eq. (11-13) requires $\mathcal{O}(N^2)$ operations. Note we may calculate the gradient of any function in the same complexity of calculating the function itself (see for example [11]). In order to compute the DE gradient with complexity of $\mathcal{O}(M \log N_{\text{kernel}})$ we could have differentiated the DE approximation derived in Sec. 4. However, the resampling is an approximation causing fluctuations in the DE value as a function of $\mathbf{W}$. This may stop MI optimization at local minima. We avoid this problem altogether by taking a different approach. Rather than differentiating an approximation based on resampling, we elect to approximate the DEs derivatives (Eq. 13) directly. We do so in a similar manner to the approximation of the DE itself. In the same way as Eq. (4) is represented by Eq. (14), Eqs. (11,12) are equivalent to

$$\Phi'(t|\hat{\mathbf{s}}_k) = f * \varphi' , \tag{21}$$

[3] We use a linear interpolation function $h(\eta) = 1 - \eta$.

[4] We used a Matlab code for fast convolution based on FFT, which had been written by Luigi Rosa, `luigi.rosa@tiscali.it`, http://utenti.lycos.it/matlab.

[5] Typically M and N_{kernel} are of the order of N or smaller. Therefore, the complexity needed is at most $\mathcal{O}(N \log N)$.

$$F'(t|\hat{\mathbf{s}}_k) = (f/\hat{p}) * \varphi'^{\text{mirror}} , \tag{22}$$

where f is given by Eq. (15), and $\varphi'^{\text{mirror}}(t) = \varphi'(-t)$. We compute the convolution (Eq. 21) in a fast way, using the array $\mathbf{v}$ (which is f, uniformly resampled by Eq. (18))

$$\hat{\Phi}'_{\text{u}} = (\mathbf{v}/N) * \varphi'_{\text{sampled}} \tag{23}$$

Finally, we interpolate Φ'_{u} to the set of points $\hat{s}_k(l)$. We do so similarly to Eq. (20).

In a somewhat analogous manner, we obtain a fast calculation of Eq. (22), as described next. First, we uniformly resample $f/\hat{p}$: similarly to Eq. (18), we define a weighted vote function $\mathbf{w}$ on the uniform grid. For each sample $\hat{s}_k(n)$ we update the this function,

$$w(m) \leftarrow \begin{cases} w(m) + h(\eta)/\hat{p}[\hat{s}_k(n)|\hat{\mathbf{s}}_k] & \text{for } \; m = m^{\#} \\ w(m) + [1 - h(\eta)]/\hat{p}[\hat{s}_k(n)|\hat{\mathbf{s}}_k] & \text{for } \; m = m^{\#} + 1 \; , \end{cases} \tag{24}$$

where $\hat{p}[\hat{s}_k(n)|\hat{\mathbf{s}}_k]$ has been computed in (20). We associate $\mathbf{w}/N$ with $f/\hat{p}$. In addition, we define a sampled version of φ'^{mirror}, termed $\varphi'^{\text{mirror}}_{\text{sampled}}$. We thus imitate Eq. (22) by

$$\hat{F}'_{\text{u}} = (\mathbf{w}/N) * \varphi'^{\text{mirror}}_{\text{sampled}} . \tag{25}$$

Finally, we interpolate F'_{u} to the set of points $\hat{s}_k(l)$, similarly to Eq. (20).

Recall from Sec. 4 that the complexity of the voting and the interpolation is $\mathcal{O}(N)$, while the complexity of the discrete convolution is $\mathcal{O}(M \log N_{\text{kernel}})$. Moreover, the complexity of Eqs. (13) is $\mathcal{O}(N)$, while the complexity of Eq. (9) is $\mathcal{O}(K^2 N)$. Thus, the overall complexity of calculating the DEs gradients of the K signals is the same as of calculating the entropy itself, $\mathcal{O}(KM \log N_{\text{kernel}} + K^2 N)$. A pseudo-code for the DE estimator and its gradient is given in [11].

6 Demonstrations

In order to evaluate our method, we performed numerous separation simulations. The first set of simulations dealt with random sources of 3K samples. We simulated six sources: four of the sources were random i.i.d., with an exponential PDF[$\alpha = 2$], an exponential PDF[$\alpha = 0.6$], a normal PDF[0,1] and a Rayleigh PDF[$\beta = 1$] (Here α and β denote the parameters of the respective PDFs [2]). The other two sources were extracted as data vectors from the Lena and Trees standard pictures. The sources were mixed using randomly generated square matrices (condition number≤ 20).

Source separation was attempted using three parametric ICA algorithms [4, 3, 6]: InfoMax, Jade and Fast ICA. In addition, separation was attempted using two non parametric ICA algorithms: the first is based on Sec. 3 and thus does not use fast convolution. The second algorithm is the one we described in Secs. 4 and 5. The software for the prior algorithms [4, 3, 6] was downloaded from the websites of the respective authors.

In order to limit the signals to the grid range we use, we first performed a rough normalization of the raw measurements. We subtracted the mean of each signal and divided it by its standard deviation. The InfoMax and FastICA

Table 1. Simulation results: The accuracy of the separation is measured in terms of the signal to interference ratio (SIR).

Algorithm	SIR [dB]	Time
Non-parametric ICA, based on Sec. 3	18 ± 4	**760** min
Non-parametric ICA with fast kernel convolution, using 1K voting bins	$\mathbf{22} \pm 3$	**1.2** min
Jade	$\mathbf{7} \pm 4$	0.2 sec
InfoMax	$\mathbf{1} \pm 0.5$	1.4 sec
InfoMax with pre-filtering	$\mathbf{8} \pm 4$	1.6 sec
Fast ICA	$\mathbf{4} \pm 4$	1.1 sec
Fast ICA with pre-filtering	$\mathbf{5} \pm 3$	1.9 sec

algorithms are more efficient when the measured signals are sparse. We thus pre-filtered the inputs to these algorithms using the derivative operator $[-1\,0\,1]/2$. Our separation procedure was based on the BFGS Quasi-Newton algorithm as implemented in the MATLAB optimization toolbox (function FMINUNC).

The results of the simulations are presented in Table 1. The separation quality is given by the signal to interference ratio (SIR)[6]. After performing numerous simulations, we report the mean SIR and the standard deviation of the SIR. Clearly, Table 1 shows that practically no degradation of the separation quality is caused by our entropy approximation. On the other hand, the improvement in the run time is huge, compared to the competing non-parametric method. Our method does not compete with the parametric algorithms over run time, but it outperforms them in separation quality. We can separate signals that the parametric methods fail to handle.

Fig. 2. Four samples of a set of 10 pictures involved in a separation simulation. The mixed signals had been filtered by a derivative operator prior to optimization. The separation SIR is 20dB.

[6] SIR=$\min_k(\|\mathbf{s}_k\|^2/\|\mathbf{s}_k - \hat{\mathbf{s}}_k\|^2)$. Note that the SIR is based on the signal k having the worst separation quality. As explained in [11], the estimated $\hat{\mathbf{s}}_k$ is prone to permutation and scale ambiguities. Thus, SIR is calculated from separation results which are compensated for these ambiguities.

To visually demonstrate the separation quality, we performed an additional set of separation simulations based on 10 pictures. The pictures were mixed using randomly generated full rank matrices. The results are presented in Fig. 2.

To Conclude: We presented an algorithm that delivers high performance and possesses low computational complexity. The low complexity makes non parametric ICA applicable to high dimensional problems and large sample sizes. We have yet to study the influence of the number of uniform grid nodes on the algorithm performance.

References

1. Anthony J. Bell and Terrence J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.
2. Riccardo Boscolo, Hong Pan, and Vwani P. Roychowdhury. Non-parametric ICA. In *Proc. ICA*, pages 13–18, 2001.
3. Jean-François Cardoso and Antoine Souloumiac. Blind beamforming for non Gaussian signals. *IEE Proceedings-F*, 140(6):362–370, 1993.
4. A. Hyvärinen. The Fast-ICA MATLAB package. 1998. http://www.cis.hut.fi/~aapo/.
5. Aapo Hyvärinen, Juha Karhunen, and Erkki Oja. *Independent component analysis*. John Wiley and Sons, USA, 2001.
6. S. Makeig, A.J. Bell, T-P Jung, and T.J. Sejnowski. Independent component analysis of electroencephalographic data. *Advances in Neural Inf. Proc. Systems 8*, pages 145–151, 1996.
7. Dinh Tuan Pham. Fast algorithm for estimating mutual information, entropies and score functions. In *Proc. ICA*, pages 17–22, 2003.
8. D.T. Pham and P Garrat. Blind separation of a mixture of independent sources through a quasi-maximum likelihood approach. *IEEE Trans. Sig. Proc.*, 45(7): 1712–1725, 1997.
9. Yoav Y. Schechner, Nahum Kiryati, and Ronen Basri. Separation of transparent layers using focus. *Int. J. Computer Vision*, 89:25–39, 2000.
10. Yoav Y. Schechner, Joseph Shamir, and Nahum Kiryati. Polarization and statistical analysis of scenes containing a semi-reflector. *J. Opt. Soc. America A*, 17:276–284, 2000.
11. Sarit Shwartz, Michael Zibulevsky, and Yoav Y. Schechner. Fast kernel entropy estimation and optimization. Technical report, No. 1431, Dep. Elec. Eng., Technion Israel Inst. Tech., 2004.
12. Paul A. Viola. *Alignment by Maximization of mutual information*. PhD thesis, MIT - Artificial Intelligence Lab., 1995.