Entropy Optimization

Application to Blind Source Separation

Anisse Taleb and Christian Jutten*

LTIRF/INPG, 46 Avenue Félix Viallet, 38000 Grenoble, France

Abstract. This paper proposes an approach for entropy optimization by neural networks. A brief introduction to this problem is given. A simple neural algorithm based upon MSE minimization is provided. Validation of this algorithm is given by an application to the Source Separation problem.

1 Introduction

Unsupervised learning algorithms aim to find hidden structures and informative representations of large data sets. The infomax principle of Linsker [8], which is a fundamental principle of self-organization, states that the transformation of a vector $\mathbf{x}$ observed on the input layer of a Neural Network (NN) to a vector $\mathbf{y}$ on the output, should be chosen in order to maximize the transinformation between input $\mathbf{x}$ and output $\mathbf{y}$.

The Exploratory Projection Pursuit (EPP) [5, 4], is a statistical technique for finding interesting structure in high dimensional data sets. When implemented by NN, EPP minimizes the output entropy to draw this one far from normal. Blind source separation (SS) and the Independent Components Analysis (ICA) [3, 6] are algorithms that learn from the input samples, an inverse matrix in order to provide statistically independent outputs : this can be achieved by minimizing output mutual information. Many other unsupervised learning algorithms use information quantities to perform their learning [10].

These examples show how entropy plays a central role in unsupervised learning. To optimize output entropy, one needs to estimate the output probability density function (pdf) or more exactly its derivates.

Such a mechanism can be done by a multilayer perceptron (MLP) in unsupervised learning, with weights vector $\mathbf{w}$. Let $\mathbf{x}$ be the input vector, and $\mathbf{y}$ the output of this MLP. Reffering to Shannon's information theory, the output entropy writes as:

$$H(\mathbf{y}) = -\int p_Y(\mathbf{y}) \log p_Y(\mathbf{y}) d\mathbf{y} = -E[\log p_Y(\mathbf{y})], \tag{1}$$

where $E[.]$ denotes the expectation operator. Weights vector $\mathbf{w}$ is trained in order to optimize (1) [2] , the stochastic gradient learning algorithm is

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mu_t \nabla_{\mathbf{w}} \mathbf{y}^T \nabla_{\mathbf{y}} \log p_Y(\mathbf{y}), \tag{2}$$

* C. Jutten is professor at ISTG in Université Joseph Fourrier of Grenoble.

[2] Generally, this is done under some constraints not explicited here.

where μ_t is the learning rate, the sign of the learning rate determines if we want to maximize or to minimize output entropy.

The purpose of this paper is to provide an efficient method for the estimation of $\nabla_{\mathbf{y}} \log p_Y(\mathbf{y})$, also called the *score functions*. In the next section we give a definition and an useful lemma and lemmas, then we describe a simple neural algorithm for the estimation of these functions. Finally, we show how this algorithm can be easily used in the SS problem.

2 Density function and Score function

Definition. Let $X = (X_1, \cdots, X_n)$ a $\mathbb{R}^n$ random variable, with differentiable pdf $p_X(\mathbf{x})$. Score function in the mutivariate case is defined as:

$$\begin{aligned} \psi_X : \mathbb{R}^n &\to \mathbb{R}^n \\ \mathbf{x} &\to \nabla_{\mathbf{x}} \log p_X(\mathbf{x}) = \left(\frac{\partial \log p_X(\mathbf{x})}{\partial x_1}, \ldots, \frac{\partial \log p_X(\mathbf{x})}{\partial x_n} \right)^T \end{aligned} \tag{3}$$

Example. For a scalar random variable X, $\psi_X(x) = \frac{p'_X(x)}{p_X(x)}$. If X is gaussian with mean m_x and variance σ_x^2, then $\psi_X(x) = -\frac{1}{\sigma_x^2}(x - m_x)$.

Lemma. *Let $X = (X_1, \cdots, X_n)$ a $\mathbb{R}^n$ random variable, with score function ψ_X, let $\mathcal{F}$ be any differentiable function of $\mathbb{R}^n \to \mathbb{R}$ satisfying:*

$$\forall i = 1 \cdots n \quad \lim_{|x_i| \to +\infty} p_X(\mathbf{x}) \mathcal{F}(\mathbf{x}) = 0, \tag{4}$$

then we have

$$E\left[\mathcal{F}(\mathbf{x}) \psi_X(\mathbf{x})\right] = -E\left[\nabla_{\mathbf{x}} \mathcal{F}(\mathbf{x})\right] \tag{5}$$

The proof is quite intuitive and consists in integrating by parts, in the scalar case the result is immediate.

Example. Following the previous example, we verify the lemma for $\mathcal{F}(x) = x$ $E[x\psi_X(x)] = E\left[-\frac{1}{\sigma_x^2}(x^2 - x m_x)\right] = -\frac{E[x^2] - m_x^2}{\sigma_x^2} = -1.$

3 Mean Square Error Minimization lead to an Unsupervised Algorithm

Scalar case Suppose now that the function $\psi_X(x)$ is available, *i.e.* known. Then, using function approximation ability of neural networks, we could use a simple MLP with one input and one output, to provide an estimation $h(\mathbf{w}, x)$ of $\psi_X(x)$. Parameters vector $\mathbf{w}$ is trained to minimize the mean squared error:

$$\mathcal{E} = \frac{1}{2} E\left[(h(\mathbf{w}, x) - \psi_X(x))^2 \right]. \tag{6}$$

A gradient descent algorithm on (6) leads to the following weights update

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mu_t \nabla_{\mathbf{w}} \mathcal{E}. \tag{7}$$

Applying the previous lemma, the gradient of the error $\mathcal{E}$ writes as:

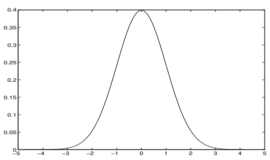
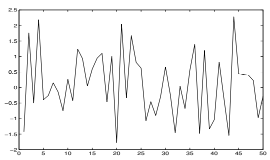
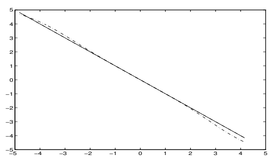
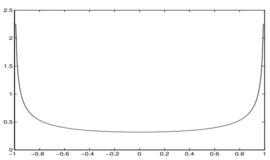
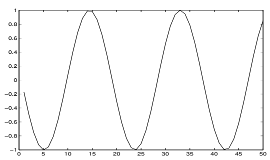
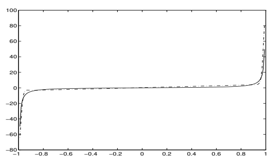
$$\nabla_{\mathbf{w}} \mathcal{E} = E\left[h(\mathbf{w}, x) \nabla_{\mathbf{w}} h(\mathbf{w}, x) + \nabla_{\mathbf{w}} \frac{\partial h(\mathbf{w}, x)}{\partial x} \right]. \tag{8}$$

In the last equation, $\psi_X(x)$ disappears: it shows surprisingly that the supervised learning algorithm for minimizing $\mathcal{E}$ does not need the *teacher* $\psi_X(x)$, and is in fact unsupervised.

Extension to the multivariate case is quite evident and is done by using a multilayer perceptron with n inputs. It leads also to an unsupervised algorithm, due to the elimination of $\psi_X(\mathbf{x})$ by using the previous lemma.

Practical issues The previous algorithms are based on a simple gradient descent. To improve the speed of convergence we suggest the use of second order minimization techniques. The algorithm was tested for scalar random variables, using a simple MLP with only one hidden layer, containing six neurons (with tanh activation function). Table 1 contains results for two distributions: the comparison between theoritical and estimated score functions proves the quality of the estimation.

Table 1. Sample experiments

distribution	Signal Wave Form	Real (–) *vs.* Estimated (---) Score Function
 Gaussian $p(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{x^2}{2})$		 $\psi(x) = -x$
 sine $p(x) = \frac{1}{\pi\sqrt{1-x^2}}$		 $\psi(x) = \frac{x}{1-x^2}$

4 Application To Blind Separation of Sources

The problem of Blind Source Separation (SS) consists in recovering the waveforms of unknown statistically independent signals called *sources*, by observing only their linear mixtures. Let us consider n independent and stationary unknown signals $\mathbf{s}(t) = (s_1(t), \cdots, s_n(t))^T$, and let $\mathbf{e}(t) = (e_1(t), \cdots, e_n(t))^T$ denoting a instantaneaous linear mixture of $\mathbf{s}$

$$\mathbf{e}(t) = \mathbf{A}\mathbf{s}(t), \tag{9}$$

where $\mathbf{A}$ is an $n \times n$ nonsingular matrix. The mixture is called instantaneous when the mixing matrix entries are time invariant scalars.

The problem is then to recover $\mathbf{s}(t)$ observing only $\mathbf{e}(t)$. This can be done by estimating a full rank matrix $\mathbf{B}$ such that

$$\mathbf{y}(t) = \mathbf{B}\mathbf{e}(t) \tag{10}$$

has statistically independent components. It is well known [3, 6] that the problem has two indeterminacies: sources can be recovered only up to a scale factor and to an permutation. In other words

$$\mathbf{y}(t) = \mathbf{P}\mathbf{D}\mathbf{s}(t) \tag{11}$$

where $\mathbf{P}$ is a permutation matrix, and $\mathbf{D}$ a diagonal matrix.

As proposed by a few authors [3, 11] (for a review and complete list of references see [7]), the independence of output components (components of $\mathbf{y}(t)$) means $\prod_i p_{Y_i}(y_i) = p_Y(\mathbf{y})$. It can be achieved by minimizing the mutual information:

$$I(\mathbf{y}) = \sum_{i=1}^{n} H(y_i) - H(\mathbf{y}), \tag{12}$$

since it is always positive and vanishes iff. independance is reached.

The joint entropy of $\mathbf{y}$ writes as $H(\mathbf{y}) = H(\mathbf{e}) + \log|det(\mathbf{B})|$: It depends only on input entropy and on the determinant of matrix $\mathbf{B}$. Using the stochastic *relative* (or natural) [1] gradient algorithm for updating matrix $\mathbf{B}$ leads to:

$$\mathbf{B}_{t+1} = (\mathbf{I} + \mu_t \mathbf{H}(\mathbf{y}_t))\mathbf{B}_t, \tag{13}$$

where $\mathbf{H}(\mathbf{y})$ is a matrix depending only on $\mathbf{y}$, with general entry

$$h_{ij}(\mathbf{y}) = \begin{cases} 0 & \text{if } i = j, \\ \psi_{Y_i}(y_i)y_j & \text{otherwise.} \end{cases} \tag{14}$$

The score functions in (14) are estimated by the algorithm described in the previous section in the one dimensional case and not *a priori* chosen like in most of SS algorithms.

The use of score functions in SS is not new. Pham *et al.* [9] proved that optimal criterion needs the knowledge of score functions. More recently, Charkani *et al.* [2] extended the results for convolutive mixtures. They both proposed

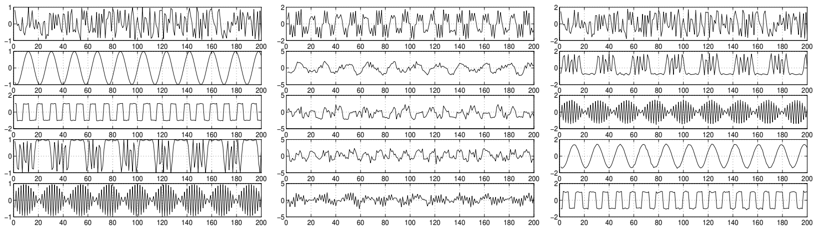


Fig. 1. Original sources, mixtures, estimated signals

suboptimal algorithm based on a parametric estimation of the score function according to a linear model:

$$\psi_{Y_i}(y_i) = \sum_k \xi_{ik} f_k(y_i), \tag{15}$$

this kind of estimation requires an appropriate choice for $f_k(.)$.

Due to the scale indeterminacy the algorithm in its previous form (13) can became unstable. In fact any scale change on the components of $\mathbf{y}$ will drive the NN to estimate a new score function: wich can imply a constant increase or decrease of the coefficients of matrix $\mathbf{B}$. A simple modification of the diagonal elements of $\mathbf{H}(\mathbf{y})$ (14), providing a normalisation to the output components of $\mathbf{y}$, avoids instability. The diagonal entries of $\mathbf{H}$ are given by

$$h_{ii}(\mathbf{y}) = \alpha_i(1 - y_i^2), \tag{16}$$

where α_i are positive scalars. This modification imposes, at convergence, that output vector $\mathbf{y}(t)$ has unit variance. This solution is preffered upon the whitening technique of Cardoso *et al.* [1], because it is less computational consuming and it can be shown that it leads to smaller rejection rates.

A sample run of this algorithm is shown in figure (1) and figure (2). Notice the low estimation variance due to the choise of optimal SS nonlinearities.

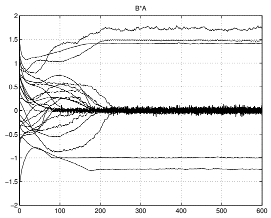


Fig. 2. Convergence of **BA** coefficients to 0 and to constant

5 Conclusion

In this paper, we presented a simple generic algorithm for the estimation of score functions. Although based on the minimization of a mean square error, which normally leads to supervised learning algorithms, this algorithm is in fact unsupervised.

Numerous algorithms use entropy optimisation learning rules, then score functions, to optimally perform the learning. For example, the application of this algorithm to SS was successful. The resulting algorithm uses adaptive on-line estimation of the score functions insteed of an *ad-hoc* one like in most SS algorithms, wich at convergence reach the optimal SS nonlinearities.

The generic nature of this algorithm allows its use in a large class of information theoritic problems. We are actually develloping new application for nonlinear source separation, image processing, Adaptive blind deconvolution and the exploratory projection pursuit.

References

1. J-F. Cardoso and B. Laheld. Equivariant adaptive source separation. *IEEE Trans. on S.P.*, 44(12):3017–3030, December 1996.
2. N. Charkani and Y. Deville. Optimization of the asymptotic performance of time-domain convolutive source separation algorithms. In *ESANN'97*, pages 273–278, Bruges, Belgium, April 1997.
3. P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36(3):287 – 314, April 1994.
4. C. Fyfe and R. Baddeley. Non-linear data structure extraction using simple hebbian networks. *Biological Cybernetics*, (72):533–541, 1995.
5. P.J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, 1985.
6. C. Jutten and J. Hérault. Blind separation of sources, Part I: An adaptive algorithm based on a neuromimetic architecture. *Signal Processing*, 24(1):1–10, 1991.
7. J. Karhunen, E. Oja, L. Wang, R. Vigário, and J. Joutsensalo. A class of neural networks for independant component analysis. *IEEE trans. N.N.*, 8(3):486–504, May 1997.
8. R. Linsker. Self-organization in a perceptual network. *Computer*, (21):105–117, 1988.
9. D. T. Pham, P. Garat, and C. Jutten. Separation of a mixture of independent sources through a maximum likelihood approach. In J. Vandewalle, R. Boite, M. Moonen, and A. Oosterlinck, editors, *Signal Processing VI, Theories and Applications*, pages 771–774, Brussels, Belgium, August 1992. Elsevier.
10. N. N. Schraudolph. *Optimization of entropy with neural networks.* PhD thesis, University of California, San Diego, 1995.
11. H.H. Yang and S.I. Amari. Adaptive on-line learning algorithms for blind separation– maximum entropy and minimum mutual information. *Neural Computation*, 1997. Accepted.