The Fixed-Point Algorithm and Maximum Likelihood Estimation for Independent Component Analysis

Aapo Hyvärinen
Helsinki University of Technology
Laboratory of Computer and Information Science
P.O.Box 5400, FIN-02015 HUT, Finland
`aapo.hyvarinen@hut.fi`
`http://www.cis.hut.fi/~aapo/`

Abstract

The author introduced previously a fast fixed-point algorithm for independent component analysis. The algorithm was derived from objective functions motivated by projection pursuit. In this paper, it is shown that the algorithm is closely connected to maximum likelihood estimation as well. The basic fixed-point algorithm maximizes the likelihood under the constraint of decorrelation, if the score function is used as the nonlinearity. Modifications of the algorithm maximize the likelihood without constraints.

Keywords: independent component analysis, blind source separation, fixed-point algorithm

1 Introduction

Independent component analysis (ICA) [10] is a statistical technique whose main applications are blind source separation, blind deconvolution, and feature extraction. In the simplest form of ICA [5], one observes m scalar random variables $x_1, x_2, ..., x_m$ which are assumed to be linear combinations of n unknown independent components, denoted by $s_1, s_2, ..., s_n$. The independent components s_i are assumed to be mutually *statistically independent*, and zero-mean. Arranging the observed variables x_j into a vector $\mathbf{x} = (x_1, x_2, ..., x_m)^T$ and the component variables s_i into a vector $\mathbf{s}$, the linear relationship can be expressed as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{1}$$

Here, $\mathbf{A}$ is an unknown $m \times n$ matrix of full column rank, called the mixing matrix. The basic problem of ICA is then to estimate both the mixing matrix $\mathbf{A}$ and the realizations of the independent componets s_i *using only observations of the mixtures* x_j.

Several estimation methods for ICA have been proposed recently. The two methods most widely used in practice seem to be the fixed-point algorithm [7, 8, 9] and the maximum likelihood (or infomax) stochastic gradient algorithm [1, 2, 3, 4]. The fixed-point algorithm was originally derived [7, 8, 9] from objective functions motivated by projection pursuit [6], and therefore its connection to estimation of the ICA data model has been rather indirect. Maximum likelihood estimation is, in contrast, the mainstream method of statistical estimation.

The purpose of this paper is to show that the fixed-point algorithm is very closely connected to maximum likelihood estimation of the ICA model. It is shown that if the nonlinearity used is chosen correctly, the fixed-point algorithm maximizes the likelihood approximately, i.e. under constraint of decorrelation. Hybrid versions between the fixed-point algorithm and the ML stochastic gradient algorithm are proposed for exact and fast maximization of the likelihood.

2 A new form of fixed-point algorithm

To begin with, we shall derive the fixed-point algorithm for *one* unit, using an objective function motivated by projection pursuit (see [8] for details). Consider one neural unit with weight vector $\mathbf{w}$ and input $\mathbf{x}$. The goal is to find the extrema of $E\{G(\mathbf{w}^T\mathbf{x})\}$ for a given non-quadratic function G, under the constraint $E\{(\mathbf{w}^T\mathbf{x})^2\} = 1$. According to the Kuhn-Tucker conditions [12], the extrema are obtained at points where

$$E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{C}\mathbf{w} = 0 \tag{2}$$

where $\mathbf{C} = E\{\mathbf{x}\mathbf{x}^T\}$, and β is a constant that can be easily evaluated to give $\beta = E\{\mathbf{w}_0^T\mathbf{x}g(\mathbf{w}_0^T\mathbf{x})\}$, where $\mathbf{w}_0$ is the value of $\mathbf{w}$ at the optimum. Let us try to solve this equation by Newton's method. Denoting the function on the left-hand side of (2) by F, we obtain its Jacobian matrix $JF(\mathbf{w})$ as

$$JF(\mathbf{w}) = E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{C} \tag{3}$$

To simplify the inversion of this matrix, we decide to approximate the first term in (3). A reasonable approximation is $E\{\mathbf{x}\mathbf{x}^T g'(\mathbf{w}^T\mathbf{x})\} = E\{\mathbf{x}\mathbf{x}^T\}E\{g'(\mathbf{w}^T\mathbf{x})\} = E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{C}$. The obtained approximation of the Jacobian matrix can be inverted easily:

$$JF(\mathbf{w})^{-1} \approx \mathbf{C}^{-1}/(E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta). \tag{4}$$

We also approximate β using the current value of $\mathbf{w}$ instead of $\mathbf{w}_0$. Thus we obtain the following approximative Newton iteration:

$$\mathbf{w}^+ = \mathbf{w} - [\mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - \beta\mathbf{w}]/[E\{g'(\mathbf{w}^T\mathbf{x})\} - \beta] \tag{5}$$

where $\mathbf{w}^+$ denotes the new value of $\mathbf{w}$, $\beta = E\{\mathbf{w}^T\mathbf{x}g(\mathbf{w}^T\mathbf{x})\}$. After every step, $\mathbf{w}^+$ is normalized by dividing it by $\sqrt{(\mathbf{w}^+)^T\mathbf{C}\mathbf{w}^+}$ to improve stability. This algorithm can be further algebraically simplified (see [8]) to obtain the original form of the fixed-point algorithm [7]:

$$\mathbf{w}^+ = \mathbf{C}^{-1}E\{\mathbf{x}g(\mathbf{w}^T\mathbf{x})\} - E\{g'(\mathbf{w}^T\mathbf{x})\}\mathbf{w}. \tag{6}$$

These two forms are equivalent. Note that for prewhitened data, $\mathbf{C}^{-1}$ disappears, giving an extremely simple form of the Newton iteration. In [9], this learning rule (for kurtosis only) was derived as a fixed-point iteration of a neural learning rule; hence the name of the algorithm.

To use the fixed-point algorithm for *several* units, the iteration in (5) is applied separately for the weight vector of each unit. Let us assume, as usual in ML estimation, that the number of independent components to be estimated equals the number of observed variables, i.e. $n = m$ and $\mathbf{A}$ is square. Denote by $\mathbf{W}$ the matrix whose columns are the weight vectors: $\mathbf{W} = (\mathbf{w}_1, ..., \mathbf{w}_n)$. Furthermore, to avoid the inversion of the covariance matrix, we can approximate it as $\mathbf{C}^{-1} \approx \mathbf{W}\mathbf{W}^T$, since $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ and $\mathbf{W}^T$ is an estimate of $\mathbf{A}$. Thus we obtain the following *novel form of the fixed-point algorithm*:

$$\mathbf{W}^+ = \mathbf{W} + \mathbf{W}[E\{\mathbf{y}g(\mathbf{y}^T)\} - \mathrm{diag}(\beta_i)]\mathbf{D} \tag{7}$$

where $\mathbf{y} = \mathbf{W}^T\mathbf{x}$, $\beta_i = E\{y_i g(y_i)\}$, and $\mathbf{D} = \mathrm{diag}(1/(\beta_i - E\{g'(y_i)\}))$. After every iteration of (7), the outputs y_i are decorrelated and normalized to unit variance; simple decorrelation algorithms that require neither matrix inversion nor eigen-value decomposition are given in [7, 8]. Note that for prewhitened data, this form is still equivalent to the original (generalized) fixed-point algorithm in [7].

3 The Fixed-point Algorithm as Maximum Likelihood Estimation

The connection between (7) and the maximum likelihood algorithm is made obscure by differences in notation. We used here (and in previous papers) the convention $\mathbf{y} = \mathbf{W}^T\mathbf{x}$ whereas in the context of maximum likelihood estimation, one defines $\mathbf{y} = \mathbf{W}\mathbf{x}$, without transpose. Let us denote $\mathbf{V} = \mathbf{W}^T$, and express (7) in the notation used in [1, 2, 3, 4]; we get

$$\mathbf{V}^+ = \mathbf{V} + \mathbf{D}[\mathrm{diag}(-\beta_i) + E\{g(\mathbf{y})\mathbf{y}^T\}]\mathbf{V}. \tag{8}$$

This is to be compared with the form of the stochastic gradient method for maximizing likelihood:

$$\mathbf{V}^{+} = \mathbf{V} + \mu[\mathbf{I} + g(\mathbf{y})\mathbf{y}^{T}]\mathbf{V}. \tag{9}$$

where μ is the learning rate, not necessarily constant in time. If g is the true score function, i.e. the derivative of the log-likelihood of the independent components (assumed to be identically distributed for simplicity), we have $-\beta_i = 1, i = 1, .., n$. Thus it is seen that *the fixed-point algorithm in Eq. (8) or (7) is essentially a batch version of the ML stochastic gradient method in (9)*, with the following differences:

1. In (8), an optimal step size is used. It is given separately for each row of $\mathbf{V}$ by the elements in $\mathbf{D}$.

2. Replacing $\mathbf{I}$ by $\text{diag}(-\beta_i)$ is also beneficial for convergence speed. Though these expressions are equal when the nonlinearity is the true score function of the independent components, in practice they are at least slightly different. This is equivalent using rescaled version of the nonlinearity g.

3. In the fixed-point algorithm, the outputs y_i are decorrelated and normalized to unit variance after every step. No such operations are needed in the gradient descent rule. The fixed-point algorithm in (8) is not stable if these additional operations are omitted.

4. The fixed-point algorithm can estimate both sub- and super-gaussian independent components. The reason is clear from (8): The matrix $\mathbf{D}$ contains estimates on the nature (sub- or super-gaussian) of the independent components. Such estimates are implicit in the simple form given in (6). For the gradient descent algorithm, the nature of the independent components has to estimated separately, as in the so-called 'extended' ICA algorithms [11].

This results suggests also how to construct a hybrid learning rule that maximizes the likelihood without constraints, yet converging faster in batch mode than simple gradient descent. This is obtained by introducing a step size coefficient α in (7):

$$\mathbf{V}^{+} = \mathbf{V} + \alpha \mathbf{D}[\mathrm{diag}(-\beta_i) + E\{g(\mathbf{y})\mathbf{y}^T\}]\mathbf{V}. \tag{10}$$

Taking a sufficiently small α (e.g. 0.1), this algorithm is stable without additional decorrelation or normalization. This algorithm maximizes likelihood almost exactly; if $\mathrm{diag}(-\beta_i)$ is further replaced by the identity matrix, it maximizes likelihood exactly. It is assumed here that the non-linearity is chosen as a good approximation of the score function, as usual in maximum likelihood estimation. Prewhitening of the data greatly improves the stability of (10). It is to be expected that (10) has faster convergence than the ordinary gradient descent rule because the step size is adapted separately for each component, as a function of the pdf's of the independent components. Moreover, the replacement of the identity matrix by the diagonal matrix in (10) should further improve convergence speed.

4 Conclusion

It was shown that the fixed-point algorithm for ICA [7, 8, 9] can be interpreted as maximizing the likelihood of the parameters of the ICA data model under the constraint of decorrelation, which is equivalent to maximizing the information flow in a neural network [2, 13]. Modifications of the fixed-point algorithm give hybrids of gradient descent and fixed-point methods, and maximize the likelihood exactly. It is thus seen that the difference between the fixed-point algorithm and the (stochastic) gradient method for maximizing likelihood [1, 2, 3, 4] is not so much in the objective function as in the method of optimization. The fixed-point algorithm provides a very fast method for maximization of likelihood in *batch* (or block) mode; it has been found in simulations to be 10–100 times faster than gradient methods [9]. The stochastic gradient methods maximizing likelihood, on the other hand, provide a simple *adaptive* learning rule that may also be biologically more plausible.

As a spin-off of these mathematical developments, we proposed a new version of the fixed-point algorithm that requires neither prewhitening nor matrix inversion.

References

[1] S.-I. Amari. Neural learning in structured parameter spaces — natural riemannian gradient. In *Advances in Neural Information Processing 9 (Proc. NIPS*96)*, pages 127–133. MIT Press, Cambridge, MA, 1997.

[2] A.J. Bell and T.J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

[3] J. F. Cardoso. Entropic contrasts for source separation. In S. Haykin, editor, *Adaptive Unsupervised Learning*, in press.

[4] A. Cichocki and R. Unbehauen. Robust neural networks with on-line learning for blind identification and blind separation of sources. *IEEE Trans. on Circuits and Systems*, 43(11):894–906, 1996.

[5] P. Comon. Independent component analysis – a new concept? *Signal Processing*, 36:287–314, 1994.

[6] P.J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, 1985.

[7] A. Hyvärinen. A family of fixed-point algorithms for independent component analysis. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP'97)*, pages 3917–3920, Munich, Germany, 1997.

[8] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. on Neural Networks*, in press.

[9] A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.

[10] C. Jutten and J. Herault. Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, 24:1–10, 1991.

[11] T.-W. Lee, M. Girolami, and T. J. Sejnowski. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. *Neural Computation*, 11:609–633, 1999.

[12] D. G. Luenberger. *Optimization by Vector Space Methods.* John Wiley & Sons, 1969.

[13] J.-P. Nadal and N. Parga. Non-linear neurons in the low noise limit: a factorial code maximizes information transfer. *Network*, 5:565–581, 1994.