Notes on the Infomax Algorithm

Upamanyu Madhow

Abstract

We briefly review the maximum likelihood interpretation of the extended Infomax algorithm for independent component analysis (ICA), including the concept of relative gradient used for iterative updates.

1 Maximum Likelihood Formulation

Consider a single snapshot of the mixing model

$$\mathbf{X} = \mathbf{AS}$$

where $\mathbf{X}$, $\mathbf{S}$ are $n \times 1$, and $\mathbf{A}$ is $n \times n$. We would like to "unmix" the sources by applying an $n \times n$ matrix $\mathbf{W}$ to get

$$\mathbf{Y} = \mathbf{WX}$$

In maximum likelihood (ML) estimation, we estimate a parameter θ based on observation $\mathbf{x}$ by maximizing the conditional density $p(\mathbf{x}|\theta)$. In order to apply this approach to estimation of $\mathbf{W}$, we must know the conditional density of $\mathbf{x}$ given $\mathbf{W}$. Given $\mathbf{W}$, we can compute $\mathbf{Y} = \mathbf{WX}$, and we apply ML estimation to this setting by assuming that we know the density of $\mathbf{Y}$. For the "right" $\mathbf{W}$, we assume that (a) the components of $\mathbf{Y}$ are independent, (b) they have known marginal densities $p_i(y_i)$, $i = 1, .., n$.

In practical terms, these marginal densities do not need to be the same as those of the actual independent components: all they do is to provide nonlinearities of the form $\frac{d}{dy_i} \log p(y_i)$ for iterative update of $\mathbf{W}$. As we have seen from our discussion of the fastICA algorithm, there are a broad range of nonlinearities that can move us towards non-Gaussianity and independence (although only the fourth order nonlinearity is guaranteed to converge to a global optimum). Thus, it makes sense that there should be some flexibility in the choice of nonlinearities in the Infomax algorithm, which is essentially similar in philosophy (except that it uses different nonlinearities and a gradient-based update rather than a Newton update).

Equating the probabilities of small volumes, we have

$$p(\mathbf{x}|\mathbf{W})|d\mathbf{x}| = p(\mathbf{y})|d\mathbf{y}|$$

Since

$$\frac{|d\mathbf{y}|}{|d\mathbf{x}|} = |det(\mathbf{W})|$$

we have

$$p(\mathbf{x}|\mathbf{W}) = p(\mathbf{y})|det(\mathbf{W})|$$

Taking the log and using the independence of the components of $\mathbf{Y}$, we obtain that the cost function to be maximized over $\mathbf{W}$ is

$$J(\mathbf{W}) = \log p(\mathbf{x}|\mathbf{W}) = \log |det(\mathbf{W})| + \sum_{i=1}^{n} \log p_i(y_i) \tag{1}$$

We would now like to adapt $\mathbf{W}$ to maximize this cost function using gradient ascent. It has been argued by Cardoso and Laheld [2] that the "right" gradient to use is one that depends on how close the outputs $\mathbf{Y}$ are to their desired distributions, rather than what combination of $\mathbf{X}$ and $\mathbf{W}$ was used to get there. This leads to the concept of *relative gradient,* reviewed next. For the setting of interest to us, this concept coincides with that of *natural gradient* introduced by Amari [3], who shows that it gives the steepest descent/ascent direction when the parameter being optimized follows a Riemannian rather than a Euclidean geometry. It is beyond our present scope to get into these arguments in detail.

2 Relative Gradient

The idea behind relative gradient is as follows: we wish to change $\mathbf{W}$ in reaction to changes in $\mathbf{Y} = \mathbf{WX}$, rather than individually to changes in $\mathbf{X}$ and $\mathbf{W}$. To this end, consider perturbations to $\mathbf{W}$ of the form

$$\Delta\mathbf{W} = \mathcal{E}\mathbf{W}$$

where $\mathcal{E}$ is a matrix with small entries. We now get

$$\begin{aligned} \mathbf{Y} + \Delta\mathbf{Y} &= (\mathbf{W} + \Delta\mathbf{W})\,\mathbf{X} = (\mathbf{W} + \mathcal{E}\mathbf{W})\,\mathbf{X} \\ &= \mathbf{Y} + \mathcal{E}\mathbf{Y} = (\mathbf{I} + \mathcal{E})\,\mathbf{Y} \end{aligned}$$

The relative gradient $\bigtriangledown_r J(\mathbf{W})$ is defined via the following equation:

$$J(\mathbf{W} + \mathcal{E}\mathbf{W}) = J(\mathbf{W}) + \langle \bigtriangledown_r J(\mathbf{W}), \mathcal{E}\rangle + o(\mathcal{E}) \tag{2}$$

where the matrix inner product used above is defined in the same manner as vector inner products (multiplying component by component and then adding):

$$\langle \mathbf{M}, \mathbf{N}\rangle = \sum_{i,j} M_{ij} N_{ij} = \mathrm{trace}(\mathbf{M}^T\mathbf{N}\rangle$$

Let us now compute the relative gradient for our particular cost function, beginning with the logarithm of the determinant:

$$\begin{aligned} &\log |det(\mathbf{W} + \mathcal{E}\mathbf{W})| - \log |det(\mathbf{W})| = \log |det(\mathbf{W} + \mathcal{E}\mathbf{W})\mathbf{W}^{-1}| \\ &\quad = \log |det(\mathbf{I} + \mathcal{E})| \end{aligned}$$

where we have used

$$det(\mathbf{MN}) = det(\mathbf{M})det(\mathbf{N})$$

and its corollary $det(\mathbf{M}^{-1}) = 1/det(\mathbf{M})$. Now, note that for small perturbations of the identity matrix, the off-diagonal terms make second order contributions to the determinant. We therefore obtain

$$\begin{aligned} &\log |det(\mathbf{I} + \mathcal{E})| = \log \Pi_i (1 + \mathcal{E}_{ii}) + o(\mathcal{E}) = \textstyle\sum_i \log(1 + \mathcal{E}_{ii}) + o(\mathcal{E}) \\ &\quad = \textstyle\sum_i \mathcal{E}_{ii} + o(\mathcal{E}) = \langle \mathbf{I}, \mathcal{E}\rangle + o(\mathcal{E}) \end{aligned}$$

Comparing with (2), we can read off that the relative gradient of $\log|det(\mathbf{W})|$ is simply $\mathbf{I}$.

Now, let us consider one of the log pdf terms, $\log p_i(y_i)$: changing $\mathbf{W}$ by $\mathcal{E}\mathbf{W}$ changes y_i by $(\mathcal{E}\mathbf{y})_i = \sum_k \mathcal{E}_{ik} y_k$. The corresponding change in the log pdf is computed as

$$\log p_i\left(y_i + (\mathcal{E}\mathbf{y})_i\right) - \log p_i(y_i) \approx \frac{\partial \log p_i(y_i)}{\partial y_i}\ (\mathcal{E}\mathbf{y})_i = \frac{\partial \log p_i(y_i)}{\partial y_i}\ \sum_k \mathcal{E}_{ik} y_k$$

Let $G(\mathbf{Y}) = \sum_i \log p_i(y_i)$ denote the sum of the log pdf terms, the change in this term is given by

$$G(\mathbf{Y} + \mathcal{E}\mathbf{Y}) - G(\mathbf{Y}) \approx \sum_i \tfrac{\partial \log p_i(y_i)}{\partial y_i} \sum_k \mathcal{E}_{ik} y_k$$

$$= \sum_{i,k} A_{ik}\mathcal{E}_{ik} = \text{trace}\left(\mathbf{A}^T\mathcal{E}\right)$$

where

$$A_{ik} = \frac{\partial \log p_i(y_i)}{\partial y_i}\ y_k$$

Comparing with (2), we see that the relative gradient of this term is the matrix $\mathbf{A}$, which we now put in more compact form. Defining the column vector

$$\phi(\mathbf{Y}) = \left(-\frac{\partial \log p_1(y_1)}{\partial y_1}, ..., -\frac{\partial \log p_n(y_n)}{\partial y_n}\right)^T$$

we have

$$\mathbf{A} = -\phi(\mathbf{Y})\mathbf{Y}^T$$

The relative gradient of the function $J(\mathbf{W})$ in (1) is therefore given by

$$\bigtriangledown_r\ J(\mathbf{W}) = \mathbf{I} - \phi(\mathbf{Y})\mathbf{Y}^T \tag{3}$$

We now choose the relative perturbation to follow the relative gradient, so that the change in $\mathbf{W}$ satisfies

$$\Delta\mathbf{W} \sim \bigtriangledown_r\ J(\mathbf{W})\mathbf{W} = \left(\mathbf{I} - \phi(\mathbf{Y})\mathbf{Y}^T\right)\mathbf{W}$$

This yields the following gradient ascent algorithm:

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \lambda_n\left(\mathbf{I} - \phi(\mathbf{Y})\mathbf{Y}^T\right)\mathbf{W}_n$$

3 Choice of nominal densities

See [1] for the choices of densities that they use for super- and sub-Gaussian sources. They choose these so as to get $\phi(y) = y + \tanh(y)$ for super-Gaussian and $\phi(y) = y - \tanh(y)$ for sub-Gaussian sources, which is particularly convenient, because we can toggle between these two cases using a single sign parameter. This yields

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \lambda_n\left(\mathbf{I} - \mathbf{K}\tanh(\mathbf{Y})\mathbf{Y}^T - \mathbf{Y}\mathbf{Y}^T\right)\mathbf{W}_n \tag{4}$$

where $\mathbf{K}$ is a diagonal matrix, with $K_{ii} = +1$ is we are trying to select a super-Gaussian source and $K_{ii} = -1$ to select a sub-Gaussian source. The sign is chosen to reinforce the trend of the current solution, as follows.

The stability condition for a gradient ascent rule of the form (3) is that

$$\mathbb{E}[\phi_i'(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}\left[y_i\phi_i(y_i)\right] > 0 \tag{5}$$

For $\phi_i(y_i) = y_i + \tanh(y_i)$ (super-Gaussian), this reduces to

$$\mathbb{E}[1 + \mathrm{sech}^2(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}[y_i^2 + y_i \tanh(y_i)] > 0$$

which simplifies to

$$\mathbb{E}[\mathrm{sech}^2(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}[y_i \tanh(y_i)] > 0$$

For $\phi_i(y_i) = y_i - \tanh(y_i)$ (sub-Gaussian), this reduces to

$$\mathbb{E}[1 - \mathrm{sech}^2(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}[y_i^2 - y_i \tanh(y_i)] > 0$$

which simplifies to

$$\mathbb{E}[\mathrm{sech}^2(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}[y_i \tanh(y_i)] < 0$$

Thus, we can set

$$K_{ii} = \mathrm{sign}\left(\mathbb{E}[\mathrm{sech}^2(y_i)]\mathbb{E}[y_i^2] - \mathbb{E}[y_i \tanh(y_i)]\right) \tag{6}$$

where, as usual, the expectations are computed by empirical averaging.

In practical terms, we would zero mean and sphere the data, and set $\mathbf{W}$ to be an arbitrary orthogonal matrix and then start updating as in (4). The idea of setting $\mathbf{K}$ according to (6) is to reinforce trends of sub- or super-Gaussianity that are already in place.

Note that the original infomax algorithm [4], which can be interpreted as corresponding to $p(y) \sim \frac{1}{\cosh y}$ (the original derivation in (7 is quite different), is given by

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \lambda_n \left(\mathbf{I} - \tanh(\mathbf{Y})\mathbf{Y}^T\right) \mathbf{W}_n \tag{7}$$

It is worth keeping in mind that, if we are only interested in sub-Gaussian sources, then the original Infomax algorithm may be quite effective, and may be faster than the extended Infomax algorithm (see Figure 5 in [1]) because it does not need to adapt the sign of the switching parameters along the diagonal of $\mathbf{K}$ in (4).

References

[1] T.-W. Lee, M. Girolami, T. J. Sejnowski, "Independent component analysis using an extended Infomax algorithm for mixed sub-Gaussian and super-Gaussian sources," *Neural Computation,* vol. 11, no. 2, 1999.

[2] J.-F. Cardoso, B. Laheld, "Equivariant adaptive source separation," *IEEE Trans. Signal Processing,* vol. 44, no. 12, pp. 3017-3030, December1996.

[3] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural Computation,* vol. 10, no. 2, pp. 251-276, February 1998.

[4] A. J. Bell, T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation,* vol. 7, pp. 1129-1159, 1995.