32 One-unit and Fixed-point ICA Algorithms with Applications

Aapo Hyvärinen, Erkki Oja, Razvan Cristescu, Patrik Hoyer, Jarmo Hurri and Kimmo Kiviluoto

The starting point of this research project on Independent Component Analysis is the development of neural learning rules for a single unit [9,11,12]. Using these learning rules, a neural unit learns to separate from a multi-dimensional input signal one of the independent components, or a direction that has certain information-theoretic properties. This research can be considered a direct logical continuation of the original work by Oja on one-unit PCA [15], and shows clearly the connection between ICA and PCA. These one-unit learning rules are especially useful in exploratory data analysis, where they can be used to find single directions in the data space that show the most interesting and independent components in the data space.
For example, assuming that the observed signal $\mathbf{x}(t)$ is whitened, or sphered, we have obtained the following very simple learning rule for separating a sub-Gaussian independent component (i.e., an independent component of negative kurtosis):

$$\Delta\mathbf{w}(t) \propto \mathbf{x}(t)g(\mathbf{w}(t)^T\mathbf{x}(t)) - \mathbf{w}(t) \tag{89}$$

where the function g is a simple polynomial: $g(u) = au - bu^3$ with $a > 1$ and $b > 0$, $\mathbf{w}$ is the weight vector a neuron, and $\mathbf{x}$ is its input. This a very simple example of learning rules that are called Hebbian (or Hebbian-like), and which constitute one of the main paradigms in neural computing. In addition to learning rule (89), which separates sub-Gaussian independent components, one needs also a learning rule for separating super-Gaussian independent components (i.e., independent components of positive kurtosis). To achieve this, we have derived another learning rule:

$$\Delta\mathbf{w}(t) \propto b\mathbf{x}(t)(\mathbf{w}(t)^T\mathbf{x}(t))^3 - a\|\mathbf{w}(t)\|^4\mathbf{w}(t). \tag{90}$$

where $a > 0$ and $b > 0$ are constants. This is also a Hebbian learning rule.
We have also developed a fast numerical method to implement the one-unit learning rules, the FastICA algorithm [10]. This method is based on a fixed-point iteration that usually speeds up the computations needed in ICA by a factor of 10 to 100. The FastICA algorithm is in a way a combination of the two preceding learning rules; the weight vector $\mathbf{w}$ is updated as follows:

$$\mathbf{w}^*(t) = E\{\mathbf{x}(\mathbf{w}(t-1)^T\mathbf{x})^3\} - 3\mathbf{w}(t-1) \tag{91}$$

$$\mathbf{w}(t) = \mathbf{w}^*(t)/\|\mathbf{w}^*(t)\| \tag{92}$$

where the expectation is, in practice, estimated using a large sample of $\mathbf{x}$ vectors. The difference from the preceding learning rules is basically that instead of using the inputs one-by-one, the FastICA algorithm first collects a batch of input data, and then uses all those data in the same learning step, in the computation of the average. The fast convergence of this fixed-point algorithm is illustrated in Figure 1, in which four images were recovered from four mixtures using altogether only 30 iterations. Another convenient property of this algorithm is that the same algorithm separates both super-Gaussian and sub-Gaussian independent components.

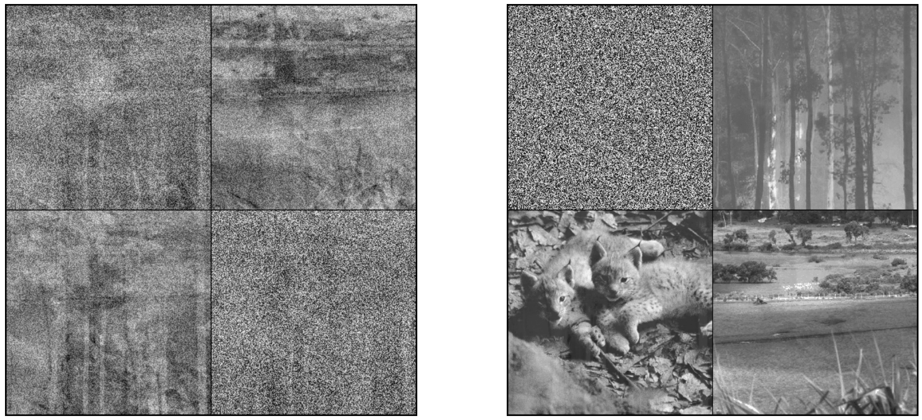

Figure 49: Three photographs of natural scenes and a noise image were linearly mixed to illustrate our algorithms. The mixtures are depicted on the left. On the right, the images recovered by the FastICA algorithm are shown. Only 7 iterations of the FastICA algorithm were required, on the average, for separating each image.

To separate several independent components, one can construct a network of several neurons, each of which learns according to the learning rules given above, and then add a feedback term to each of those learning rules.
The above learning rules, which are based on finding the extrema of kurtosis, have also been generalized for a large class of criteria of non-Gaussianity [12,2,5,9]. This means that the cubic non-linearity in the learning rules can be replaced by almost any other non-linearity. (Of course, some other changes are then also necessary.) Thus one obtains algorithms that often perform the ICA decomposition in a much more reliable and accurate way, according to such statistical criteria as asymptotic variance and robustness [3,9]. Also, the FastICA algorithm can be modified so that it works even when the data is corrupted by Gaussian noise [7].
The development of the FastICA algorithm has enabled us to apply ICA on data sets that are of a very high dimension. One example is image processing [1,8], where the FastICA algorithm has been used with success. Taking small windows of ordinary, real-life photographs, we decomposed the images into small components whose occurence is as independent from each other as possible. Some image components are depicted in Figure 2. Such a decomposition is likely to have interesting applications in image data compression, pattern recognition, and other domains of image processing. There are two reasons for this. First, such a decomposition resembles closely a so-called sparse coding. In sparse coding, one finds a coding method for the data that has certain interesting statistical properties, and fits with some neurophysiological measurements on the neural processing of sensory data. Second, the components found are reminiscent of so-called wavelets, which are used in some highly efficient techniques for image compression.
Based on the features given by ICA, we have developed a new method for image denoising [13,4] . This is based on modeling the noisy data by a noisy version of the ICA data model, and then estimating the original image by maximum likelihood estimation of the model. This results in the application of a (soft) thresholding operator on the features described above. Figure 3 shows an example of the application of this method, called sparse code shrinkage. The advantage of this method

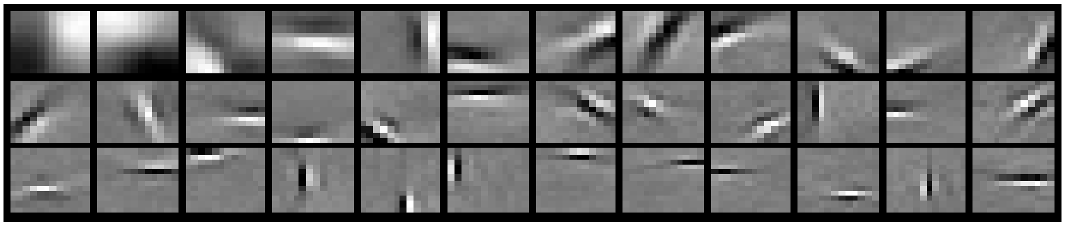

Figure 50: Image feature extraction by ICA. The FastICA algorithm was applied on image data, this time using small sub-windows of images as the data. Thus images were decomposed into hypothetical components whose occurrences are rather independent from each other. This figure shows some such components. One can see that the components define certain features that are often quite local, and resemble bar or edge detectors.

Figure 51: Image denoising by sparse code shrinkage. Using the theory of noisy ICA, one we have developed an image denoising method. This leads to a thresholding of the coefficients of the wavelet-like features shown in Figure 2.

over wavelet methods is that it is completely adaptive: both the features and the involved thresholding (shrinkage) functions are adapted to the statistical properties of the data.
We have also applied these methods on analysis of financial time series [14]. We used data that represented the simultaneous cash flow at several stores belonging to the same retail chain. ICA detected factors that affect the cash flow of all the stores. When the effect of these "fundamental factors" is removed, the impact of the actions of the management became more visible.

References

[1] J. Hurri, A. Hyvärinen, J. Karhunen, and E. Oja. Image feature extraction using independent component analysis. In *Proc. NORSIG'96*, pages 475–478, Espoo, Finland, 1996.

[2] 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.

[3] A. Hyvärinen. One-unit contrast functions for independent component analysis: A statistical analysis. In *Neural Networks for Signal Processing VII (Proc. IEEE Workshop on Neural Networks for Signal Processing)*, pages 388–397, Amelia Island, Florida, 1997.

[4] A. Hyvärinen. Independent component analysis in the presence of gaussian noise by maximizing joint likelihood. *Neurocomputing*, 22:49–67, 1998.

[5] A. Hyvärinen. New approximations of differential entropy for independent component analysis and projection pursuit. In *Advances in Neural Information Processing Systems 10*, pages 273–279. MIT Press, 1998.

[6] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. on Neural Networks*, 1999. To appear.

[7] A. Hyvärinen. Gaussian moments for noisy independent component analysis. *IEEE Signal Processing Letters*, 1999. To appear.

[8] A. Hyvärinen, R. Cristescu, and E. Oja. A fast algorithm for estimating overcomplete ICA bases for image windows. In *Proc. Int. Joint Conf. on Neural Networks*, Washington, D.C., 1999. submitted.

[9] A. Hyvärinen and E. Oja. Simple neuron models for independent component analysis. *Int. Journal of Neural Systems*, 7(6):671–687, 1996.

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

[11] A. Hyvärinen and E. Oja. One-unit learning rules for independent component analysis. In *Advances in Neural Information Processing Systems 9*, pages 480–486. MIT Press, 1997.

[12] A. Hyvärinen and E. Oja. Independent component analysis by general nonlinear Hebbian-like learning rules. *Signal Processing*, 64(3):301–313, 1998.

[13] A. Hyvärinen, E. Oja, P. Hoyer, and J. Hurri. Image feature extraction by sparse coding and independent component analysis. In *Proc. Int. Conf. on Pattern Recognition (ICPR'98)*, pages 1268–1273, Brisbane, Australia, 1998.

[14] K. Kiviluoto and E. Oja. Independent component analysis for parallel financial time series. In *Proc. ICONIP'98*, volume 2, pages 895–898, Tokyo, Japan, 1998.

[15] E. Oja. A simplified neuron model as a principal component analyzer. *J. of Mathematical Biology*, 15:267–273, 1982.