8

Overcomplete Independent Component Analysis Algorithms and Applications

When the number of sources is greater than the number of sensors, we refer to the blind source separation (BSS) problem as being overcomplete (Lee *et al*., 1999). Lewicki and Sejnowski (2000) suggest that, if some information is known in advance about the source distribution, independent components (ICs) can be extracted to some extent by the overcomplete independent component analysis (ICA) model. Theis *et al*. (2002, 2004) proposed a geometric algorithm with two steps, namely, matrix recovery and source signal recovery. They improved the geometric algorithm for mixing matrix estimation when the number of source signals is equal to the number of mixed signals, and then applied the maximum likelihood method for source signal recovery. To improve the convergence properties of the algorithm, Li, Cichocki, and Amari (2003) proposed a sparse matrix factorization method for BSS. Davies and Mitianoudis (2004) proposed a simple hybrid model to solve the problem in the modified discrete cosine transform (MDCT) processing of an audio signal. Thus, in this chapter, we introduce the classic overcomplete ICA algorithm, the algebraic overcomplete ICA (AICA) algorithm, and the geometric overcomplete ICA (Geo-ICA) algorithm.

8.1 Overcomplete ICA Algorithms

8.1.1 Classic Overcomplete ICA Algorithm

Lewicki and Olshausen (1999) and Lewicki and Sejnowski (2000) proposed an algorithm based on statistical models to solve the overcomplete problem. This algorithm consists of a linear programming method and the natural gradient method. Given

Blind Source Separation: Theory and Applications, First Edition. Xianchuan Yu, Dan Hu and Jindong Xu.
©2014 Science Press. All rights reserved. Published 2014 by John Wiley & Sons Singapore Pte. Ltd.

observation data $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n]^{\mathrm{T}}$ mixed with additive noise, the model can be expressed as

$$\boldsymbol{X} = \boldsymbol{AS} + \varepsilon \tag{8.1}$$

where A is an $n \times m(m > n)$ matrix. We usually assume that ε is the Gaussian additive noise, so

$$\log P(X|\boldsymbol{A}, \boldsymbol{S}) \propto -\frac{1}{2\sigma^2}(\boldsymbol{X} - \boldsymbol{AS})^2 \tag{8.2}$$

where σ^2 denotes the variance of the noise. Estimation of the source signal, based on a probabilistic model, attempts to find S which maximizes the posterior probability.

$$\hat{s} = \arg \max_S P(\boldsymbol{S}|\boldsymbol{X}, \boldsymbol{A}) = \arg \max_S P(\boldsymbol{X}|\boldsymbol{A}, \boldsymbol{S})P(\boldsymbol{S}) \tag{8.3}$$

where $\hat{s}$ is the estimation of S.

Given a matrix of the base vectors A and the observed data X, we can use the gradient descent method to optimize the logarithmic posterior probability distribution. In the case of an overcomplete model, the mixing matrix is not a square matrix, nor is it an inversion.

An alternative method, based on the assumption that $\varepsilon = 0$, is to solve the problem as a linear programming problem (Chen, Donoho, and Saunders, 1998):

$$\min\ \boldsymbol{c}^{\mathrm{T}}|\boldsymbol{S}|\ s.t.\ \boldsymbol{AS} = \boldsymbol{X} \tag{8.4}$$

Let $\boldsymbol{c}^{\mathrm{T}} = [1, \dots, 1]$, so $\boldsymbol{c}^{\mathrm{T}}|s| = \sum_k |s_k|$. By separating the positive and negative coefficients, Equation 8.4 can be transformed into a standard linear programming problem (with only positive coefficients). By executing the following transformations, $s \leftarrow [u; v]$, $\boldsymbol{c} \leftarrow [1; 1]$, and $\boldsymbol{A} \leftarrow [\boldsymbol{A}; -\boldsymbol{A}]$, Equation 8.4 becomes

$$\min\,[1; 1]^{\mathrm{T}}[u; v] s.t. [\boldsymbol{A}; -\boldsymbol{A}][u; v] = x,\ u, v \geq 0 \tag{8.5}$$

Equation 8.5 can be solved by a standard linear programming method, with solutions for u and v, while s is calculated by the following relationship.

$$s \leftarrow [u; v]\ s = u - v \tag{8.6}$$

To obtain a learning algorithm for A, we first need to determine an appropriate objective function; a natural goal is to maximize the probability of the observed data. For $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_n]^{\mathrm{T}}$,

$$P(\boldsymbol{X}|\boldsymbol{A}) = \prod_{k=1}^{n} P(\boldsymbol{X}_k|\boldsymbol{A}) \tag{8.7}$$

$P(X_k|\boldsymbol{A})$ can be obtained by marginalization of the internal state,

$$P(\boldsymbol{X}_k|\boldsymbol{A}) = \int \mathrm{d}sP(s)P(\boldsymbol{X}_k|\boldsymbol{A}, \boldsymbol{s}),\ k = 1, \dots, n. \tag{8.8}$$

Lewicki and Sejnowski (2000) used a multivariate Gaussian distribution on s to solve the above equation approximatively, and obtain the learning rule. This is given as

$$\Delta \boldsymbol{A} \propto \boldsymbol{A}\boldsymbol{A}^{\mathrm{T}} \frac{\partial}{\partial \boldsymbol{A}} \log P(x|\boldsymbol{A}) \approx -\boldsymbol{A}(z\widehat{s}^{\mathrm{T}} + \boldsymbol{I}) \tag{8.9}$$

where $z_k = \partial \log P(s_k)/\partial s_k$. Note that the learning rule in Equation 8.9 and standard ICA based on the natural gradient method are very similar in form.

8.1.2 Algebraic Overcomplete ICA Algorithm

See Waheed and Salam, 2002, 2003a, 2003b.

The idea behind the AICA and Geo-ICA algorithms is basically the same. The first step involves learning the projection of the matrix of base vectors A on the unit circle, while the second step restores the source vector.

The AICA algorithm calculates the distance between vectors using the dot product. The dot product between vectors α and β can be defined as

$$\alpha \bullet \beta = \alpha^{\mathrm{T}}\beta = |\alpha||\beta| \cos \theta_{\alpha\beta} \tag{8.10}$$

where $\theta_{\alpha\beta}$ represents the angle between vectors α and β. If α and β represent the projection of the two different vectors on the unit circle, the distance between them can be characterized completely by the direction angle $\theta_{\alpha\beta}$. Another benefit of using the dot product is that this type of operation is commutative, so $\alpha \bullet \beta = \beta \bullet \alpha$. As a result of the above characteristics, the AICA algorithm does not require the symbol transformation and replacement operation as in the Geo-ICA algorithm when calculating the distance of the base vector. Thus, the AICA algorithm only needs to calculate n dot product operations instead of $2^n - 1$ Euclidean distance calculations as is the case in the Geo-ICA algorithm.

The first step in the algorithm is to calculate the matrix of the base vectors, the implementation of which is described below.

Algorithm AICA algorithm: calculating the matrix of the base vectors

Input: Sample set $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_N]$
Output: Matrix of base vectors $\boldsymbol{A}_{n\times m}$
Steps:

1. Select m vectors in an n-dimensional unit circle to make up the weight matrix $\boldsymbol{A}_0 = [a_1, a_2, \ldots, a_m]$. When the matrix is initialized, an initial point can be selected randomly on the unit circle or by following a particular distribution.
2. Select an n-dimensional sample $\boldsymbol{X}_i$ from the observation vector group, making $\boldsymbol{X}_i \neq 0, i = 1, 2, \ldots, n$.
3. Selected sample $\boldsymbol{X}_i$ is projected onto the unit circle, such that $\boldsymbol{Y}_i = \frac{\boldsymbol{X}_i}{||\boldsymbol{X}_i||}$, $i = 1, 2, \ldots, n$.

4. Calculate the dot product between the column vectors of $\boldsymbol{Y}_i$ and $\boldsymbol{A_i}$.
5. $a_j^* = \underset{a_j}{\text{argmax}} \, |\boldsymbol{Y}_i^{\mathrm{T}} \cdot a_j|, i = 1, 2, \dots, n; j = 1, 2, \dots, m.$
6. Apply iterative updating according to the nearest principle.

$$a_j(k+1)^* = \begin{cases} \psi\left(a_j(k)^* + \eta(t)\text{sgn}(\boldsymbol{Y}_i - a_j(k)^*)\right), \boldsymbol{Y}_i^{\mathrm{T}} \cdot a_j^* > 0 \\ \psi(a_j(k)^* + \eta(t)\text{sgn}(-\boldsymbol{Y}_i - a_j(k)^*)), \boldsymbol{Y}_i^{\mathrm{T}} \cdot a_j^* < 0 \end{cases}$$

   where $\eta(t)$ is the learning rate dependent on the number of iterations, similar to that in the Geo-ICA algorithm; $\psi(\cdot)$ is the nonlinear equation used to project a column vector in the weight matrix onto the unit circle.
7. Update the weight matrix.

$$\Delta\boldsymbol{A}_i = \boldsymbol{A}_i - \boldsymbol{A}_{i-1}, i = 1, 2, \dots, n$$

8. If $|\Delta\boldsymbol{A_i}| < \varepsilon$ $(\varepsilon > 0)$, the algorithm converges to its end state, finally yielding $\boldsymbol{A}_i$, which is the requested matrix of the base vectors.
9. If the algorithm does not satisfy the convergence conditions, repeat steps (2)–(7) until convergence is achieved.

The second step in the algorithm is source vector recovery. The process can use the shortest path algorithm, which is also used in the Geo-ICA algorithm, or the linear programming algorithm that is used in the overcomplete ICA algorithm.

8.1.3 Geometric Overcomplete ICA Algorithm

See Theis *et al*., 2002, 2004.

The idea behind this algorithm is explained below. To obtain $\boldsymbol{A}$ and $\boldsymbol{S}$ in an overcomplete own learning process is quite difficult; however, one could first find $\boldsymbol{A}'$, which is the $\boldsymbol{A}$ projection on the unit circle. Then $\boldsymbol{S}$ can be approximately obtained through this. The Geo-ICA algorithm includes two steps: matrix recovery and source signal recovery.

The matrix recovery process is described below.

1. Choose $2m$ unit roots $\omega_1, \omega'_1, \dots, \omega_m, \omega'_m$ on the unit circle $\boldsymbol{S}^{n-1} \subset \boldsymbol{R}^n$, making the ω_i and ω'_i symbols opposite and paired, that is, $\omega_i = -\omega'_i$ $(i = 1, \dots, m)$. These ω_i are usually called neurons.
2. Fix the learning rate $\eta : N \to R$, making $\eta(t) > 0$, $\sum_{m \in N} \eta(m) = \infty$, and $\sum_{m \in N} \eta(m)^2 < \infty$.
3. Iterate by executing the following steps until the convergence conditions are satisfied. Select a sample $\boldsymbol{X}(t) \in \boldsymbol{R}^n$ of random variable $\boldsymbol{X}$; if $\boldsymbol{X}(t) = 0$, reselect a nonzero sample. By assuming that the probability distribution function of $\boldsymbol{X}$ is a continuous function, it could happen that the sample value is zero. $\boldsymbol{X}(t)$ is projected

onto the unit circle, then $\boldsymbol{Y}(t) = \frac{X(t)}{||X(t)||}$. Let ω_i or ω'_i be the nearest point from $\boldsymbol{Y}(t)$ calculated by the Euclidean distance, and execute the following iteration

$$\omega_i(t+1) = \pi(\omega_i(t) + \eta(t)\pi(\boldsymbol{Y}(t) - \omega_i(t))) \tag{8.11}$$

where $\pi : \boldsymbol{R}^n \backslash 0 \to \boldsymbol{S}^{n-1}$ represents the projection from $\boldsymbol{R}^n$ to the $n-1$-dimensional unit circle $\boldsymbol{S}^{n-1}$, as well as

$$\omega'_i(t+1) = -\omega_i(t+1) \tag{8.12}$$

4. All neurons are updated by these iterations.

This algorithm can be considered the absolute winner-takes-all learning algorithm. It is similar to a special case of Kohonen's competitive learning algorithm for self-organizing maps, which is a zero-neighborhood algorithm. The difference is that the step in the sample direction in the former algorithm does not depend on the distance, and the learning process is conducted on the unit circle $\boldsymbol{S}^{n-1}$, rather than on $\boldsymbol{R}^{n-1}$.

After completion of the above iterative algorithm, we get the group $\omega_1, \omega'_1, \dots, \omega_m, \omega'_m, \omega_1, \dots, \omega_m$ composed of matrix $\boldsymbol{A}$, which is the next operation.

The source signal recovery process uses the shortest path algorithm. For simplicity, we assume $n = 2$. The goal of the algorithm is $\text{argmin}_{\boldsymbol{x}_\lambda = \boldsymbol{As}} |\boldsymbol{S}|_1$, where $|S|_1$ denotes the L1 paradigm of $\boldsymbol{S}$, and mixing matrix $\boldsymbol{A}$ can be obtained by the first step in the Geo-ICA algorithm. Let $\boldsymbol{a}_k (k = 1, \dots, m)$ represent a column vector after matrix $\boldsymbol{A}$ is normalized. Theis *et al.* (2002) presented the following conclusions. For any given sample $\boldsymbol{x}_i$, identify the two column vectors $\boldsymbol{a}_j$ and $\boldsymbol{a}_k$ of matrix $\boldsymbol{A}$ that are the closest to $\boldsymbol{x}_i$; then $\boldsymbol{S}_i \in \boldsymbol{R}^m$ can be expressed as

$$(\boldsymbol{S}_i)_l = \begin{cases} \left(\left(a_j | a_k \right)^{-1} x_i \right)_j, l = j \\ ((a_j | a_k)^{-1} x_i)_k, l = k \\ \quad 0, \text{otherwise} \end{cases} \tag{8.13}$$

It is not difficult to prove that $\boldsymbol{A}$ and $\boldsymbol{S}_i$ obtained by the above operation satisfy $\boldsymbol{AS}_i = \boldsymbol{X}_i$.

8.2 Applications and Analysis

Most of the classic face recognition algorithms are subspace analysis methods. For example, the principal component analysis (PCA) and linear discriminant analysis (LDA) algorithms obtain a facial image by depicting the input data from a high-dimensional space to a low-dimensional subspace. Subspace refinement typically requires a massive facial database. However, in many practical application fields the available data for refining are limited, or only a small number of the resulting refined images are available. To improve recognition rate we must obtain as much information as possible from a limited number of input data. From the foregoing, compared with the number of obtained observation signals, overcomplete ICA can obtain more source signals. This section focuses on overcomplete ICA in facial feature extraction.

8.2.1 Overcomplete ICA Facial Feature Representation

A face image can be represented as a linear superposition of a group of independent base images. If x_i represents a face image, a training set comprising n training images is given as $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n]^{\mathrm{T}}$. Next, each image is whitened. Training images can be considered to be linear combinations of m independent components $\boldsymbol{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_m)$. If $m = n$, the problem becomes an ICA feature extraction problem (Hyvärinen, 2001). However, the more general case is $m > n$, that is, the overcomplete case. $\boldsymbol{X} = \boldsymbol{AS}$ represents the relationship between the training samples and ICs; in other words, each image $\boldsymbol{X}_i$ can be represented as a linear combination of $\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_m$, with its coefficients being the ith row vector of weight matrix $\boldsymbol{A}$, as shown in Figure 8.1 (Jiang and Zhu, 2005).

Finally, with the coordinate vectors, $(a_{i1}, a_{i2}, \ldots, a_{im})$ as the overcomplete ICA face representation, such an image can be expressed using the base image coordinates. Overcomplete ICA provides a better face representation method, since overcomplete basis vectors offer more facial features to represent all the face data.

8.2.2 Experiments and Conclusions

We used some of the face images in the FERET database for our experiments (Cheng *et al*., 2003). First, we randomly selected 20 images from the test database, and then extracted the required data from these 20 images. The test database contained a total of 420 images of 70 individuals. Six 112×92 images, each of which differed with respect to illumination and expression, were included for each individual.

Using the model described in Equation 8.1, $\boldsymbol{X} = \boldsymbol{AS} + \varepsilon$, where $\boldsymbol{X}$ is a $20 \times 10\,304$ matrix with the first row comprising a facial image for processing. $\boldsymbol{A}$ is a 20×30 matrix. The source signal matrix $\boldsymbol{S}$ is obtained as a 30×10304 matrix using the overcomplete ICA algorithm described earlier. Each row of $\boldsymbol{S}$ is a source image. Figure 8.2 illustrates the 30 source images.

In our experiments, we compared overcomplete ICA, PCA, and standard ICA. Identification of facial features and a comparison of the recognition rate using PCA (Turk and Pentland, 2001) and standard ICA (Bartlett, Movellan, and Sejnowski,

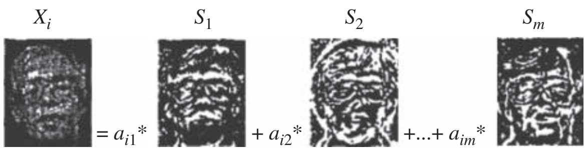

Figure 8.1 Representation of an overcomplete ICA human face

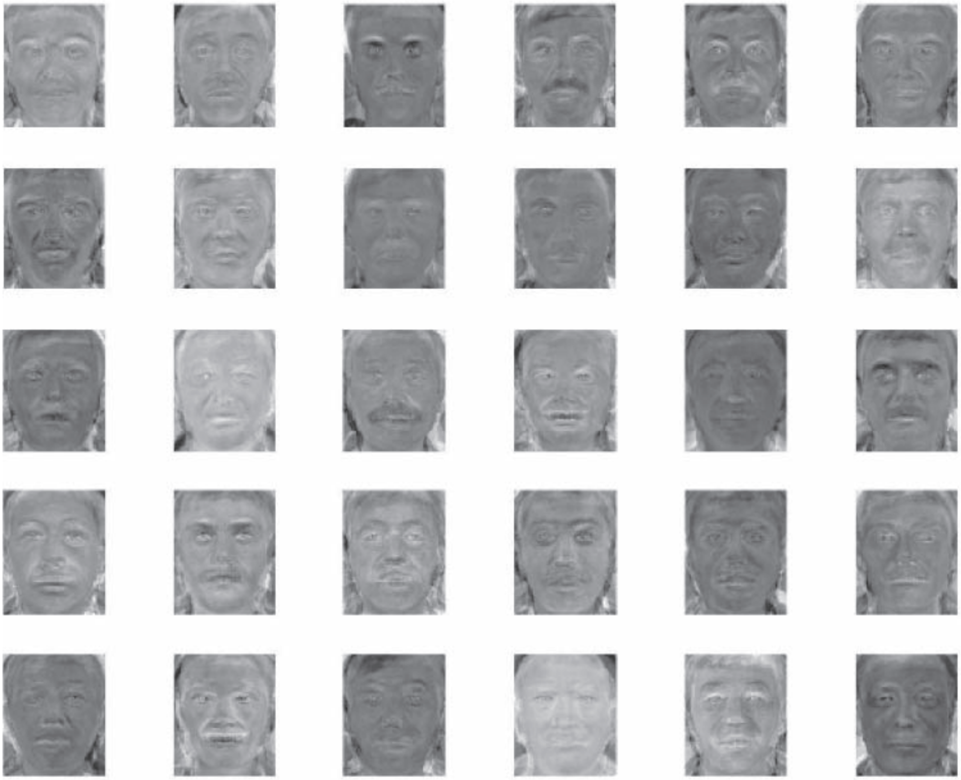

Figure 8.2 Source images

2002) was carried out using the same database and test set. The facial images extracted by overcomplete ICA, PCA, and standard ICA are referred to as the overcomplete IC face, the eigen face, and the IC face. These facial images were projected as a basic feature set $\boldsymbol{B}$, while each image from the test database was also projected onto the basic feature set $\boldsymbol{B}$. Coefficient f is a feature vector used to identify the face

$$f = \boldsymbol{B}a \tag{8.14}$$

The similarity measure was calculated as the distance between the feature vectors f, thereby evaluating the effect of the face recognition experiment: a smaller distance between the two vectors denotes greater similarity. The distance formula is calculated as

$$d_{ij} = \frac{f_i \cdot f_j}{\|f_i\| \cdot \|f_j\|} \tag{8.15}$$

The first and second rows in Figure 8.3 (Jiang and Zhu, 2005) show the results of PCA feature selection from the face samples, while the third and fourth rows show the

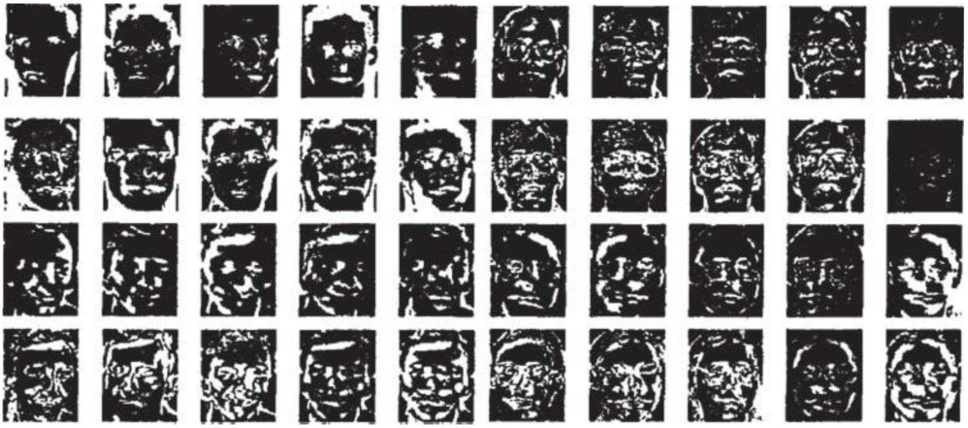

Figure 8.3 Eigen faces and IC faces

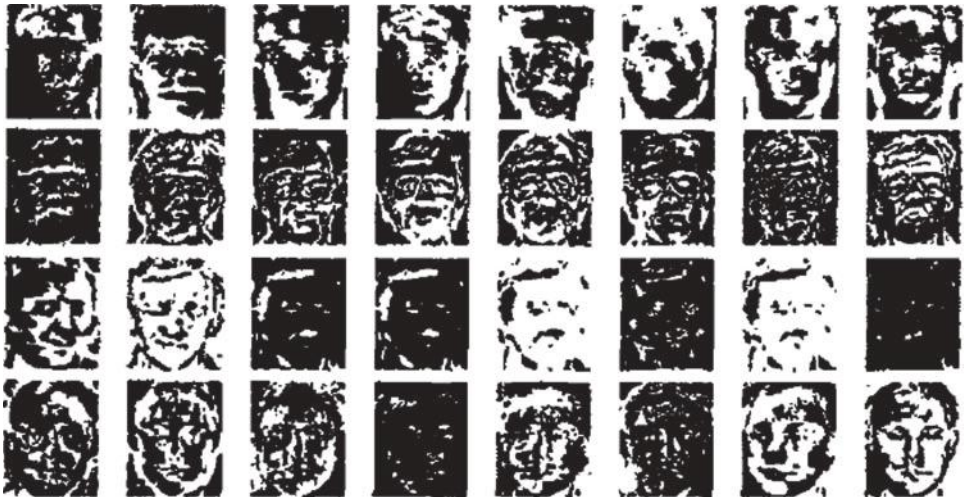

Figure 8.4 Overcomplete IC faces

results of ICA feature selection. In other words, a sample training face that participated in the learning is mapped to the subspace spanned by the IC and principle component using the ICA and PCA methods, respectively. Figure 8.4 shows the overcomplete IC extracted by overcomplete ICA.

In effect, the eigen faces extracted by PCA retain the basic shape of the face; whereas the IC faces extracted by ICA depict the contour lines of the face better, and reflect local statistical characteristics of the faces and image intensity. The overcomplete IC faces extracted by overcomplete ICA are able to characterize the facial contours better, and fully reflect the “sparse” characteristics. The overcomplete ICA algorithm

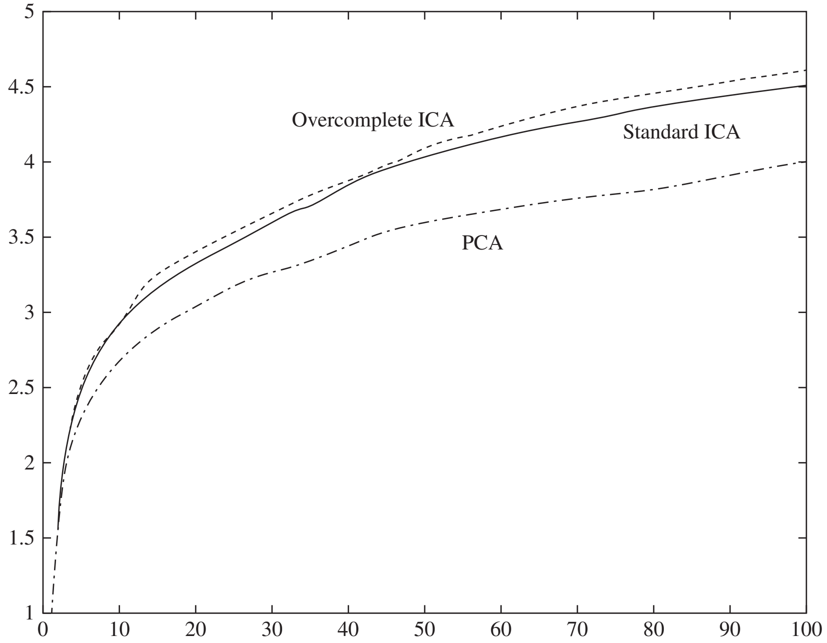

Figure 8.5 Comparison of experimental results using overcomplete ICA, basic ICA, and PCA

introduces some noise in the calculation process, which has some effect on the overcomplete eigen faces.

Figure 8.5 shows the comparison results of the face recognition experiment using the three methods: overcomplete ICA, basic ICA, and PCA. According to Figure 8.5, overcomplete ICA and basic ICA achieve better results than the PCA algorithm, while compared with basic ICA, overcomplete ICA offers certain improvements. Based on these results, use of overcomplete ICA for face recognition is beneficial, and if the feature extraction algorithm is used to further improve the performance, the advantage is even greater.

8.3 Chapter Summary

When the number of ICs is greater than the number of observed mixed signals, we refer to the BSS problem as being overcomplete. Similar to the case of noise ICA, overcomplete ICA has to solve two different problems, that is, how to estimate the mixing matrix and the ICs. This is completely different to ordinary ICA, because in general ICA these two problems can be addressed simultaneously.

If the basis vectors are overcomplete, it is more difficult to describe their likelihood, because the problem belongs to the class of problems with missing data. One approach for solving this problem is to use more prior knowledge about the density and time structure of the source, while another is to take advantage of the sparse representation of the data matrix. By first estimating the mixing matrix, and then the source, if the source is sufficiently sparse, it can be separated directly. The AICA and Geo-ICA algorithms described in this chapter adopt the latter approach.

References

Bartlett, M.S., Movellan, J.B., and Sejnowski, T.J. (2002) Face recognition by independent component analysis. *IEEE Transactions on Neural Networks*, **13**(6), 1450–1464.

Chen, S.S., Donoho, D.L., and Saunders, M.A. (1998) Atomic decomposition by basis pursuit. *Scientific Computing*, **20**(1), 33–61.

Cheng, J., Lu, H.Q., Chen, Y.W., and Zeng, X.Y. (2003) Face recognition using overcomplete independent component analysis. *Knowledge-Based Intelligent Information and Engineering Systems*, **27**(3), 1443–1448.

Davies, M., and Mitianoudis, N. (2004) Simple mixture model for sparse overcomplete ICA. *IEE Proceedings-Vision Image and Signal Processing*, **151**(1), 35–43.

Hyvärinen, A. (2001) Blind source separation by nonstationarity of variance: a cumulate based approach. *IEEE Transactions on Neural Networks*, **12**(6), 1471–1474.

Jiang, Y.W., and Zhu, S.M. (2005) Face feature extraction based on overcomplete ICA feature extraction. *Computer Science*, **32**(7), 162–165, (in Chinese).

Lee, T., Lewicki, M.S., Girolami, M., and Sejnowski, T.J. (1999) Blind source separation of more sources than mixtures using overcomplete representations. *IEEE Transactions on Signal Processing*, **6**(4), 87–90.

Lewicki, M.S., and Olshausen, B.A. (1999) Probabilistic framework for the adaptation and comparison of image codes. *Journal of the Optical Society of America A: Optics Image Science and Vision*, **16**(7), 1587–1601.

Lewicki, M.S., and Sejnowski, T.J. (2000) Learning overcomplete representations. *Neural Computation*, **12**(2), 337–365.

Li, Y., Cichocki, A., and Amari, S.I. (2003) Sparse component analysis for blind source separation with less sensors than sources. Proceedings of ICA-2003, pp. 89–94.

Theis, F.J., Lang, E.W., and Puntonet, C.G. (2004) A geometric algorithm for overcomplete linear ICA. *Neural Computation*, **56**(1), 1–18.

Theis, F.J., Lang, E.W., Westenhuber, T., and Puntonet, C.G. (2002) Overcomplete ICA with a geometric algorithm, in *Artificial Neural Networks – ICANN 2002*, Vol. **2415**, Springer, Berlin, pp. 1049–1054.

Turk, M., and Pentland, A. (1991) Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, **3**(1), 77–86.

Waheed, K., and Salam, F.M. (2002) State-space blind source recovery for mixtures of multiple source distributions. *IEEE International Symposium on Circuits and Systems*, **1**, 197–200.

Waheed, K., and Salem, F.M. (2003) Algebraic independent component analysis: an approach for separation of overcomplete speech mixtures. *Proceedings of the International Joint Conference on Neural Networks*, **1**, 775–780.

Waheed, K., and Salam, F.M. (2003b) Algebraic overcomplete independent component analysis. Circuits, Proceedings of ICA-2003, pp. 1–6.