1. Overall convey paper contribution

A fundamental problem in neural network research, as well as in many other disciplines, is finding a suitable representation of multivariate data, i.e. random vectors. For reasons of computational and conceptual simplicity, the representation is often sought as a linear transformation of the original data. In other words, each component of the representation is a linear combination of the original variables. Well-known linear transformation methods include principal component analysis, factor analysis, and projection pursuit. Independent component analysis (ICA) is a recently developed method in which the goal is to find a linear representation of nongaussian data so that the components are statistically independent, or as independent as possible. Such a representation seems to capture the essential structure of the data in many applications, including feature extraction and signal separation. In this paper, we present the basic theory and applications of ICA, and our recent work on the subject.

1. What paper talks about

In this review, we cover the definition and underlying principles of ICA in Sections 2 and 3. Then, starting from Section 4, the ICA problem is solved on the basis of minimizing or maximizing certain conrast functions; this transforms the ICA problem to a numerical optimization problem. Many contrast functions are given and the relations between them are clarified. Section 5 covers a useful preprocessing that greatly helps solving the ICA problem, and Section 6 reviews one of the most efficient practical learning rules for solving the problem, the FastICA algorithm. Then, in Section 7, typical applications of ICA are covered: removing artefacts from brain signal recordings, finding hidden factors in financial time series, and reducing noise in natural images. Section 8 concludes the text.

1. Relevant material

PCA want to fond the one that best explain the variability of the data

ICA want to find the one in which each vector is an independent component of data

1. Cocktail party problem

Imagine that you are in a room where two people are speaking simultaneously. You have two microphones, which you hold in different locations. The microphones give you two recorded time signals, which we could denote by x1(t) and x2(t), with x1 and x2 the amplitudes, and t the time index. Each of these recorded signals is a weighted sum of the speech signals emitted by the two speakers, which we denote by s1(t) and s2(t). We could express this as a linear equation:

where a11,a12,a21, and a22 are some parameters that depend on the distances of the microphones from the speakers. It would be very useful if you could now estimate the two original speech signals s1(t) and s2(t), using only the recorded signals x1(t) and x2(t). This is called the cocktail-party problem.

As an illustration, consider the waveforms in Fig. 1 and Fig. 2. These are, of course, not realistic speech signals, but suffice for this illustration. The original speech signals could look something like those in Fig. 1 and the mixed signals could look like those in Fig. 2. The problem is to recover the data in Fig. 1 using only the data in Fig. 2. Actually, if we knew the parameters ai j, we could solve the linear equation in (1) by classical methods. The point is, however, that if you don’t know the ai j, the problem is considerably more difficult. One approach to solving this problem would be to use some information on the statistical properties of the signals si(t) to estimate the aii. Actually, and perhaps surprisingly, it turns out that it is enough to assume that s1(t) and s2(t), at each time instant t, are statistically independent. This is not an unrealistic assumption in many cases, and it need not be exactly true in practice. The recently developed technique of Independent Component Analysis, or ICA, can be used to estimate the ai j based on the information of their independence, which allows us to separate the two original source signals s1(t) and s2(t) from their mixtures x1(t) and x2(t). Fig. 3 gives the two signals estimated by the ICA method. As can be seen, these are very close to the original source signals (their signs are reversed, but this has no significance.)

1. ICA model

To rigorously define ICA (Jutten and Hérault, 1991; Comon, 1994), we can use a statistical “latent variables” model. Assume that we observe n linear mixtures x1,...,xn of n independent components

We have now dropped the time index t; in the ICA model, we assume that each mixture xj as well as each independent component sk is a random variable, instead of a proper time signal. The observed values xj(t), e.g., the microphone signals in the cocktail party problem, are then a sample of this random variable. Without loss of generality, we can assume that both the mixture variables and the independent components have zero mean: If this is not true, then the observable variables xi can always be centered by subtracting the sample mean, which makes the model zero-mean.

The statistical model in Eq. 4 is called independent component analysis, or ICA model. The ICA model is a generative model, which means that it describes how the observed data are generated by a process of mixing the components si . The independent components are latent variables, meaning that they cannot be directly observed. Also the mixing matrix is assumed to be unknown. All we observe is the random vector x, and we must estimate both A and s using it. This must be done under as general assumptions as possible.

The starting point for ICA is the very simple assumption that the components si are statistically independent. Statistical independence will be rigorously defined in Section 3. It will be seen below that we must also assume that the independent component must have nongaussian distributions. However, in the basic model we do not assume these distributions known (if they are known, the problem is considerably simplified.) For simplicity, we are also assuming that the unknown mixing matrix is square, but this assumption can be sometimes relaxed, as explained in Section 4.5. Then, after estimating the matrix A, we can compute its inverse, say W, and obtain the independent component simply by:

To illustrate the ICA model in statistical terms, consider two independent components that have the following uniform distributions:

The range of values for this uniform distribution were chosen so as to make the mean zero and the variance equal to one, as was agreed in the previous Section. The joint density of s1 and s2 is then uniform on a square. This follows from the basic definition that the joint density of two independent variables is just the product of their marginal densities (see Eq. 10): we need to simply compute the product. The joint density is illustrated in Figure 5 by showing data points randomly drawn from this distribution.

Now let as mix these two independent components. Let us take the following mixing matrix:

This gives us two mixed variables, x1 and x2. It is easily computed that the mixed data has a uniform distribution on a parallelogram, as shown in Figure 6. Note that the random variables x1 and x2 are not independent any more; an easy way to see this is to consider, whether it is possible to predict the value of one of them, say x2, from the value of the other. Clearly if x1 attains one of its maximum or minimum values, then this completely determines the value of x2. They are therefore not independent.

1. FastICA

In the preceding sections, we introduced different measures of nongaussianity, i.e. objective functions for ICA estimation. In practice, one also needs an algorithm for maximizing the contrast function, for example the one in (25). In this section, we introduce a very efficient method of maximization suited for this task. It is here assumed that the data is preprocessed by centering and whitening as discussed in the preceding section.

To begin with, we shall show the one-unit version of FastICA. By a "unit" we refer to a computational unit, eventually an artificial neuron, having a weight vector w that the neuron is able to update by a learning rule. The FastICA learning rule finds a direction, i.e. a unit vector w such that the projection w T x maximizes nongaussianity. Nongaussianity is here measured by the approximation of negentropy J(w T x) given in (25). Recall that the variance of w T x must here be constrained to unity; for whitened data this is equivalent to constraining the norm of w to be unity.

The basic form of the FastICA algorithm is as follows:

Note that convergence means that the old and new values of w point in the same direction, i.e. their dot-product is (almost) equal to 1. It is not necessary that the vector converges to a single point, since w and −w define the same direction. This is again because the independent components can be defined only up to a multiplicative sign. Note also that it is here assumed that the data is prewhitened.

The one-unit algorithm of the preceding subsection estimates just one of the independent components, or one projection pursuit direction. To estimate several independent components, we need to run the one-unit FastICA algorithm using several units (e.g. neurons) with weight vectors w1,...,wn. To prevent different vectors from converging to the same maxima we must decorrelate the outputs w T 1 x,...,w T n x after every iteration. We present here three methods for achieving this. A simple way of achieving decorrelation is a deflation scheme based on a Gram-Schmidt-like decorrelation. This means that we estimate the independent components one by one. When we have estimated p independent components, or p vectors w1,...,wp, we run the one-unit fixed-point algorithm for wp+1, and after every iteration step subtract from wp+1 the “projections” w T p+1wjwj , j = 1,..., p of the previously estimated p vectors, and then renormalize wp+1: