INDEPENDENT COMPONENT ANALYSIS FOR FINANCIAL TIME SERIES

Erkki Oja, Kimmo Kiviluoto, and Simona Malaroiu

Helsinki University of Technology,
Espoo, Finland
{Erkki.Oja, Kimmo.Kiviluoto, Simona.Malaroiu}@hut.fi

ABSTRACT

The recent work in the author's research group on using Independent Component Analysis (ICA) for the analysis and prediction of financial time series is reviewed. ICA belongs to the group of linear transform methods, with the goal to make a transform from the observed signals into a signal space in which the signals are statistically independent. Sometimes independence can be attained, especially in blind source separation in which the observed signals are assumed to be linear mixtures of independent source components. Then the goal of ICA is to invert the unknown mixing operation. Even when independence is not possible, as is often the case in financial time series, the ICA transformation produces useful component signals whose dependence is reduced, and that are nongaussian with a density allowing sparse coding. The ICA transformation is also related to the temporal structure of the found signals as measured by Kolmogorov complexity or its approximations. The signals are structured and hence may be easier to interpret and predict.

After discussing the ICA criterion within the context of linear signal transforms, the FastICA algorithm is reviewed. It is a computationally efficient method for finding a subset or all of the component signals. Then, two financial applications are covered: decomposition of parallel financial time series of weekly sales into basic factors, and prediction of mixture time series by linear combinations of predictions on the ICA components.

1. INTRODUCTION

Our starting point is a signal vector $\mathbf{x}(t) = (x_1(t), ..., x_m(t))^T$, consisting of the intensities or values of m time series $x_i(t)$ at time t. Suppose $\mathbf{x}(t)$ is linearly transformed into another signal vector $\mathbf{s}(t)$ of dimension n by

$$\mathbf{s}(t) = \mathbf{L}\mathbf{x}(t). \tag{1}$$

An example of such a linear transform is the Karhunen-Loeve Transform (KLT; see e.g. [21]). There, the rows of matrix $\mathbf{L}$, say $\mathbf{l}_j$, are the orthonormal eigenvectors of the data covariance matrix $\mathbf{C} = E\{\mathbf{x}(t)\mathbf{x}(t)^T\}$, where we also assume that the signals or time series are stationary and normalized to zero means, $E\{\mathbf{x}(t)\} = 0$. The result is that the elements of the transformed vector $\mathbf{s}(t) = (s_1(t), ..., s_n(t))^T$, $s_j(t) = \mathbf{l}_j^T\mathbf{x}(t)$ are uncorrelated at each time t. In a financial application, the new time series $s_j(t)$ are uncorrelated linear combinations of the original time series; if $x_i(t)$ give values of stocks, then $s_j(t)$ would be uncorrelated portfolios.

Underlying the KLT compression is the statistical technique of Principal Component Analysis (PCA). This, as well as Factor Analysis and some related techniques, is widely used in statistics. They share the property of being based on first and second order statistics of data only.

Recently, the higher-order statistical technique of Independent Component Analysis (ICA) has attracted considerable interest in the signal processing and neural network communities [1, 4, 5, 6, 7, 17, 18, 22]. It can be explained starting from PCA: instead of requiring that the elements $s_j(t)$ are uncorrelated, the goal is now to find a linear transform by which they are *statistically independent*. For Gaussian random variables, these properties are equivalent, and so it is meaningful to apply ICA only to non-Gaussian signals.

The fundamental restriction of the ICA model is that we can only estimate non-Gaussian independent components (except if just one of the independent components is Gaussian). Moreover, neither the energies nor the signs of the independent components can be estimated. For mathematical convenience, we define here that the independent component signals $s_j(t)$ have unit variance. This makes the (non-Gaussian) independent components unique, up to their signs. Note that no order is defined between the independent components.

The basic memoryless ICA model assumes that the $\mathbf{x}(t) = (x_1(t), ..., x_m(t))^T$, for different t, are just identically distributed samples from a random vector $\mathbf{x}$. Therefore, the ICA theory is usually developed using the random vector notation, and the explicit time index is dropped. We follow this convention.

It turns out that ICA is related to some other approaches, especially Projection Pursuit, Sparse Coding, and Minimum Complexity / Minimum Description Length Coding. In Projection Pursuit [9], the goal is to find linear projections s_j through the data that are somehow interesting; a typical index or contrast function for this property is deviation from the Gaussian density. Each row $\mathbf{l}_j$ of matrix $\mathbf{L}$ in eq. (1) would give one of the linear projections $s_j = \mathbf{l}_j^T\mathbf{x}$ of zero-mean data. Some criteria used in ICA in fact produce Projection Pursuit solutions if the independence assumption cannot be satisfied [18].

In Sparse Coding [3], the goal is to represent the signal vector $\mathbf{x}$ in the inverse transformation

$$\mathbf{x} = \mathbf{A}\mathbf{s} = \sum \mathbf{a}_j s_j \tag{2}$$

0-7803-5800-7/00$10.00 ©2000 IEEE

by as few components in the sum as possible. The vectors $\mathbf{a}_j$ are the columns of the matrix $\mathbf{A}$ that is called the mixing matrix. The sparsest possible code is the vector code by which exactly one of the components only appears in the sum for any given $\mathbf{x}$. Sparse coding means that most of the s_j values should be zero or very small; another way to characterize this is that the density of s_j, understood as a random variable, should be sharply peaked around zero with only a little of the probability mass in the tails. Some ICA criteria lead to such densities, often called super-Gaussian densities [15].

Yet another connection is Minimum Complexity: the complexity or regularity of a time domain signal can be theoretically expressed by Kolmogorov complexity. Recently, it was shown [24] that when the Kolmogorov complexity is approximated by compressing the signal with lossless compression and by defining the complexity to be the size of the compressed signal in bits, then efficient signal separation can be achieved even for signals that are far from independent. It was also shown that minimizing the mutual information of the sources is just a special case of minimizing the Kolmogorov complexity.

2. THE FASTICA ALGORITHM

The basic approach to solving the ICA problem is given by the information theoretic or entropic contrast functions, often based on mutual information or negentropy (see [1, 4, 5, 6, 25] and references therein). According to eq. (1), the solution is sought in the form

$$\hat{\mathbf{s}} = \mathbf{y} = \mathbf{W}\mathbf{x}. \tag{3}$$

The goal is now to find a matrix $\mathbf{W}$ that makes the elements of $\mathbf{y}$ statistically independent. We call such a matrix a separating matrix.

The usual starting point is differential entropy

$$H(\mathbf{y}) = -\int p(\mathbf{y}) log p(\mathbf{y}) d\mathbf{y}$$

where $p(\mathbf{y})$ is the density of $\mathbf{y}$. The Kullback-Leibler divergence between $p(\mathbf{y})$ and the product of its marginal densities, also called mutual information between the elements of $\mathbf{y}$, is given by

$$I(y_1, ..., y_n) = \sum_{i=1}^{n} H(y_i) - H(\mathbf{y}).$$

It is non-negative and becomes zero when y_i are independent. Mutual information is therefore a theoretically suitable contrast function for independence; however, usually we cannot assume that we know the density $p(\mathbf{y})$.

Negentropy is defined as

$$J(\mathbf{y}) = H(\mathbf{y}_{gauss}) - H(\mathbf{y})$$

where $\mathbf{y}_{gauss}$ is a Gaussian random vector with the same covariance as $\mathbf{y}$. It is known that for uncorrelated y_i, it holds

$$I(y_1, ..., y_n) = J(\mathbf{y}) - \sum_{i=1}^{n} J(y_i).$$

This is another contrast function: to maximize the sum of negentropies, we may try to find variables y_i that are as non-Gaussian as possible. When the density is unknown, approximations must be used. Many of these result in maximizing contrast functions of the form

$$J_{gen}(\mathbf{W}) = \sum_{i=1}^{n} E\{h_i(y_i)\} \tag{4}$$

with $h_i(.)$ some appropriate functions; a simple example is

$$h_i(y_i) = y_i^4.$$

This contrast maximizes kurtosis in the normalized case $E\{y_i^2\} = 1$ and is a measure of non-Gaussianity.

A recent review of the relations of the various information theoretic criteria, like mutual information, negentropy, maximum entropy, and infomax, as well as the maximum likelihood approach, is given by Cardoso [5], who also discusses the numerical problems in minimizing / maximizing such contrast functions. One alternative to solving the separating matrix $\mathbf{W}$ is to use gradient-based learning rules.

The problem of solving the separating matrix $\mathbf{W}$ is somewhat simplified if we consider only one of the source signals at a time. From eq. (3) it follows

$$\hat{s}_i = y_i = \mathbf{w}_i^T \mathbf{x}$$

with $\mathbf{w}_i$ the i-th row of $\mathbf{W}$. We have earlier suggested and analyzed neural type one-unit learning rules [12, 14] that give as solutions one row $\mathbf{w}_i$ of the separating matrix. A condition of local convergence to a correct solution was given in [14]. The condition is very robust and shows that a wide range of nonlinear functions in our learning rules are possible; this question is also discussed in [23].

The problem is further simplified by performing a preliminary sphering or prewhitening of the data $\mathbf{x}$ [7]: the observed vector $\mathbf{x}$ is first linearly transformed to another vector whose elements are mutually uncorrelated and all have unit variance. Thus its correlation matrix equals unity. This transformation is always possible and can be accomplished by classical Principal Component Analysis. At the same time, the dimensionality of the data should be reduced so that the dimension of the transformed data vector equals n, the number of independent components. This also has the effect of reducing noise. Let us assume that this prewhitening has already been done and our signal vectors $\mathbf{x}$ satisfies $E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{I}$. But then the true separating matrix is an orthogonal matrix due to our assumptions on the components s_i: in eq. (3) it holds $E\{\mathbf{y}\mathbf{y}^T\} = \mathbf{W}E\{\mathbf{x}\mathbf{x}^T\}\mathbf{W}^T = \mathbf{W}\mathbf{W}^T$, and this must be the unit matrix because we want our separated signals to be independent and have unit variances. Thus we have reduced the problem of finding an arbitrary full-rank matrix $\mathbf{W}$ to the simpler problem of finding an orthogonal matrix. This requirement is quite helpful in developing the one-unit learning rules. It also brings the ICA transformation to the domain of orthogonal transforms.

Let us now consider the general ICA contrast (4). In a gradient type learning rule, one row $\mathbf{w}_i$ of the separating matrix $\mathbf{W}$ would be sought using an instantaneous version

of the gradient

$$\begin{aligned}\frac{\partial J_{gen}}{\partial \mathbf{w}_i} &= E\{\frac{\partial h_i(y_i)}{\partial y_i}\frac{\partial y_i}{\partial \mathbf{w}_i}\} \\ &= E\{h_i'(y_i)\mathbf{x}\} \\ &= E\{h_i'(\mathbf{w}_i^T\mathbf{x})\mathbf{x}\},\end{aligned}$$

with $h_i'(.)$ the derivative of $h_i(.)$. In the instantaneous gradient, the expectation is dropped and the gradient is computed separately for each input vector $\mathbf{x}$. In addition, a normalization term would be needed that keeps the norm of $\mathbf{w}_i$ equal to one - remember that our $\mathbf{W}$ matrix must be orthogonal due to the prewhitening of the data $\mathbf{x}$.

The convergence of gradient algorithms can be proven using the principles of stochastic approximation. The advantage of such on-line learning rules is that the inputs $\mathbf{x}$ can be used in the algorithm at once, thus enabling fast adaptation in a non-stationary environment. A resulting trade-off, however, is that the convergence is slow, and depends on a good choice of the learning rates. A bad choice of the learning rate can in practice destroy convergence. Therefore, some ways to make the learning radically faster and more reliable may be needed. The fixed-point iteration algorithms are such an alternative. The fixed points $\mathbf{w}$ of any learning rule are obtained by taking the expectations and equating the change in the weight to 0. A deterministic iteration can be formed from this equation by a number of ways, e.g. by standard numerical algorithms for solving such equations.

As an example of a fixed point learning rule, consider the simple case of maximizing the kurtosis $E\{y_i^4\} - 3E\{y_i^2\}$ of the estimated signals y_i; but because we assumed that the estimated signals have unit variance, this reduces to maximizing the fourth moment $E\{y_i^4\}$. Its gradient with respect to $\mathbf{w}_i$ is $4E\{(\mathbf{w}_i^T\mathbf{x})^3\mathbf{x}\}$. Using a suitable normalizing term, we arrive at the following iteration [13]:

1. Take a random initial vector $\mathbf{w}(0)$ of norm 1. Let $k = 1$.
2. Let $\mathbf{w}(k) = E\{\mathbf{x}(\mathbf{w}(k-1)^T\mathbf{x})^3\} - 3\mathbf{w}(k-1)$. The expectation can be estimated using a large sample (say, 1000) of $\mathbf{x}$ vectors.
3. Divide $\mathbf{w}(k)$ by its norm.
4. If $|\mathbf{w}(k)^T\mathbf{w}(k-1)|$ is not close enough to 1, let $k = k+1$ and go back to step 2. Otherwise, output the vector $\mathbf{w}(k)$.

The final vector $\mathbf{w}(k)$ given by the algorithm equals one of the rows of the (orthogonal) separating matrix $\mathbf{W}$.

To estimate n independent components, we run this algorithm n times. To ensure that we estimate each time a different independent component, we only need to add a simple orthogonalizing projection inside the loop. Recall that the rows of the separating matrix $\mathbf{W}$ are orthonormal because of the sphering. Thus we can estimate the independent components one by one by projecting the current solution $\mathbf{w}(k)$ on the space orthogonal to the rows of the separating matrix $\mathbf{W}$ previously found. Also a symmetrical orthogonalization is possible [16].

This algorithm with the preliminary whitening is implemented in the FastICA package available through the World Wide Web [8]. In addition to the option of kurtosis maximization / minimization, the algorithm allows the use of general nonlinear functions $h_i(y_i)$ according to the criterion (4). A widely used choice is to use the nonlinearity $tanh(\mathbf{w}(k-1)^T\mathbf{x})$ in the algorithm instead of the cubic term $(\mathbf{w}(k-1)^T\mathbf{x})^3$; then also the normalizing term must be adjusted accordingly to maintain the fast convergence [16].

A remarkable property of the FastICA algorithm is that a very small number of iterations, usually 5-10, seems to be enough to obtain the maximal accuracy allowed by the sample data. This is due to the cubic convergence shown in [13]. A comparison of the speed and accuracy obtained with the FastICA algorithm and some other ICA algorithms was recently reported [10, 11]. It turns out that over a variety of problems with artificial and real data, the FastICA algorithm is reliable and converges about 1 to 2 decades faster than some other popular algorithms.

3. FINDING HIDDEN FACTORS IN FINANCIAL DATA

It is a tempting alternative to try ICA on financial data. There are many situations in that application domain in which parallel time series are available, such as currency exchange rates or daily returns of stocks, that may have some common underlying factors. ICA might reveal some driving mechanisms that otherwise remain hidden. In a study of a stock portfolio [2], it was found that ICA is a complementary tool to PCA, allowing the underlying structure of the data to be more readily observed. If one could find the maximally independent mixtures of the original stocks, i.e. portfolios, this might help in minimizing the risk in the investment strategy.

In [19], we applied ICA on a different problem: the cashflow of several stores belonging to the same retail chain, trying to find the fundamental factors common to all stores that affect the cashflow data. Thus, the cashflow effect of the factors specific to any particular store, i.e., the effect of the actions taken at the individual stores and in its local environment could be analyzed.

We start by considering a set of parallel time series $x_i(t)$, with i indexing the individual time series, $i = 1, \ldots, m$ and t denoting discrete time. In our case these signals are the financial time series. As explained in Section 1, the basic ICA assumes a generative model, by which the original signals $x_i(t)$ are instantaneous linear mixtures of independent source signals or underlying factors $s_j(t)$, $j = 1, \ldots, n$ with some unknown mixing coefficients $a_{i,j}$:

$$x_i(t) = \sum_j a_{i,j}s_j(t), \tag{5}$$

for each signal $x_i(t)$. We assume the effect of each time-varying underlying factor $s_j(t)$ on the measured time series to be approximately linear. Going to a vector-matrix formulation gives the ICA model (2).

The assumption of having some underlying independent components in this specific application may not be unrealistic. For example, factors like seasonal variations due to holidays and annual variations, and factors having a sudden effect on the purchasing power of the customers like prize

changes of various commodities, can be expected to have an effect on all the retail stores, and such factors can be assumed to be roughly independent of each other. Yet, depending on the policy and skills of the individual manager like e.g. advertising efforts, the effect of the factors on the cash flow of specific retail outlets are slightly different. By ICA, it is possible to isolate both the underlying factors and the effect weights, thus also making it possible to group the stores on the basis of their managerial policies using only the cash flow time series data.

The data consisted of the weekly cash flow in 40 stores that belong to the same retail chain; the cash flow measurements cover 140 weeks. Some examples of the original data $x_i(t)$ are shown in Fig. 1.

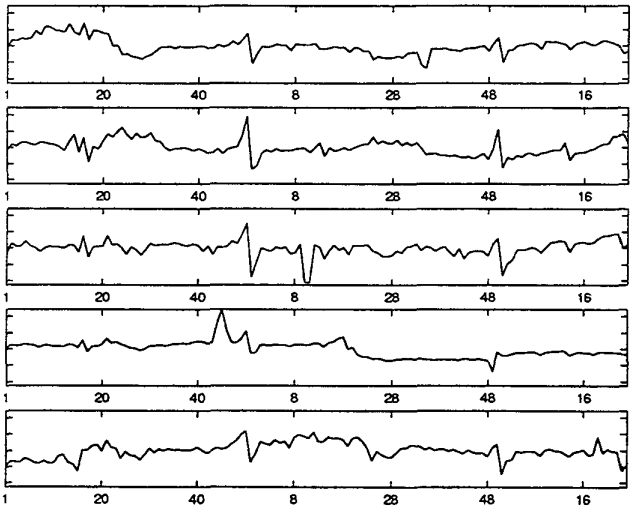

Figure 1: (from [19]). *Five samples of the fourty original cashflow time series (mean removed, normalized to unit standard deviation). Horizontal axis: time in weeks.*

The prewhitening was performed so that the original signal vectors were projected to the subspace spanned by their first four principal components and the variances were normalized to 1. Thus the dimension of the signal space was decreased from 40 to 4. Using the FastICA algorithm, four IC's $s_i(t)$, $i = 1, ..., 4$ were estimated. As depicted in Fig. 2, the FastICA algorithm has found several clearly different fundamental factors hidden in the original data.

The factors have clearly different interpretations. The upmost factor follows the sudden changes that are caused by holidays etc.; the most prominent example is the Christmas time. The factor on the bottom row, on the other hand, reflects the slower seasonal variation, with the effect of the summer holidays clearly visible. The factor on the third row could represent a still slower variation, something resembling a trend. The last factor, on the second row, is different from the others; it might be that this factor follows mostly the relative competitive position of the retail chain with respect to its competitors, but other interpretations are also possible.

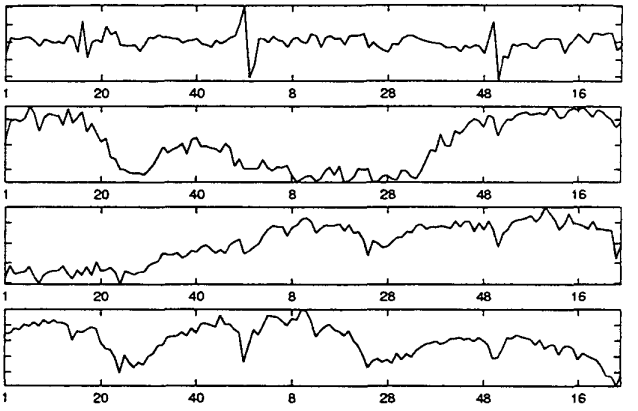

Figure 2: (from [19]). *Four independent components or fundamental factors found from the cashflow data.*

More details on the experiments and their interpretation can be found in [19].

4. TIME SERIES PREDICTION BY ICA

As already noted in the Introduction, the ICA transformation tends to produce component signals $s_j(t)$ that can be compressed with fewer bits than the original signals $x_i(t)$. They are thus more structured and regular. This gives motivation to try to predict the signals $x_i(t)$ by first going to the ICA space, doing the prediction there, and then transforming back to the original time series. The prediction can be done separately and with a different method for each component, depending on its time structure. Hence, some interaction from the user is needed in the overall prediction procedure. Another possibility would be to formulate the ICA contrast function in the first place so that it includes the prediction errors.

The basic procedure we are suggesting here is as follows [20]:

1. After subtracting the mean of each time series and removing the second order statistic effects by normalization (after which each time series has zero mean and unit variance), we estimate the independent components $s_j(t)$ of the given set of time series $x_i(t)$, and simultaneously find the mixing coefficients $a_{i,j}$ in (5). We used the FastICA package [8] implementation of the above algorithm. The number of IC's can be variable.
2. For each component, we first apply suitable linear or nonlinear filtering to reduce the effects of noise – smoothing for components that contain very low frequencies (trend, slow cyclical variations), and high-pass filtering for components containing high frequencies and/or sudden shocks. The nonlinear smoothing is formally done by applying a function f on the source vectors s(t),

$$\mathbf{s}^s(t) = \mathbf{f}[\mathbf{s}(\cdot)] \qquad (6)$$

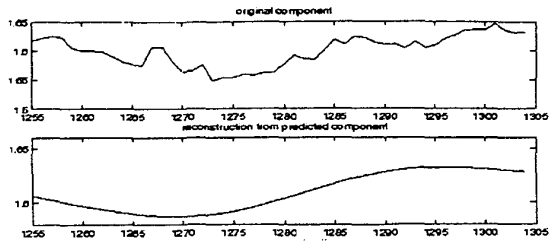

Figure 3: Prediction of real-world/financial data: the upper figure represents the original mixtures and the lower one the forecast obtained using ICA prediction for an interval of 50 values.

3. We predict each smoothed independent component separately, for instance using some method of AR-modeling [26]. The basic prediction equation is:

$$\mathbf{s}^p(t+1) = \mathbf{g}[\mathbf{s}^s(t), \mathbf{s}^s(t-1), \ldots, \mathbf{s}^s(t-q)] \tag{7}$$

4. We combine the predictions for each independent component by weighing them with the coefficients $a_{i,j}$, thus obtaining the predictions for the original observed time series $x_i(t)$:

$$\mathbf{x}^p(t) = \mathbf{A}\mathbf{s}^p(t) \tag{8}$$

To test the method, we applied our algorithm on a set of 10 relevant foreign exchange rate time series. The results were promising, as the ICA prediction performed better than direct prediction. Fig. 3 shows an example of prediction using our method. The upper figure represents one of the original mixtures and the lower one the forecast obtained using ICA prediction for an interval of 50 values. In Table 1 there is a comparison of errors obtained by applying classical AR prediction to the original time series directly, and our method outlined above. The last column shows the magnitude of the errors when no smoothing is applied to the currencies.

Table 1: The errors (in units of 0.001) obtained with our method and the classical AR method. Ten currencies were considered and five independent components.

	Errors						
Tolerances for AR prediction	2	0.5	0.1	0.08	0.06	0.05	0
ICA prediction	2.3	2.3	2.3	2.3	2.3	2.3	2.3
AR prediction	9.7	9.1	4.7	3.9	3.4	3.1	4.2

5. DISCUSSION

In many application fields like biomedicine, economy, industry, telecommunications, etc. there are situations in which multivariate time series, or several parallel signals, occur and some relevant information is buried in them. Often it is known that these hidden factors are statistically independent, because they emanate from very different causes. We often assume that they have an approximately linear effect on the measured signals, but there is no detailed model available that would give the dependencies and help solve the inverse problem. Blind signal processing techniques are needed for signal decomposition, and Independent Component Analysis (ICA) is a powerful tool in these applications. The algorithm propounded here, the FastICA algorithm based on the fixed-point learning rule, seems to be very fast and offers an appealing alternative compared to the other methods.

In Section 3, we applied ICA to financial time series data. The data is parallel, representing the simultaneous cash flow at several stores belonging to the same retail chain. The ICA detects a few factors that affect the cash flow of all the stores, although each store responds to these factors in a slightly different manner. When the effect of these fundamental factors is removed, the impact of the actions of the management becomes more visible. Additionally, it is possible to examine the differences between the stores on the basis of their responses to the fundamental factors obtained with ICA.

In Section 4, ICA was used for prediction of financial time series. The prediction algorithm performed well both on toy data and exchange rate time series. Again, we start by supposing that there are some independent factors that affect the time evolution of such time series. The economic indicators, interest rates and psychological factors can be the underlying factors of exchange rates, as they are closely tied to the evolution of the currencies. Given the time series, by forecasting the underlying factors, which in our case are the independent components, a better prediction of the time series can be obtained. The algorithm predicts very well the turning points.

ICA and AR prediction are linear techniques, however the smoothing is responsible for the non-linearity of the model. After the preprocessing with ICA, the source estimates are easily predicted. Introducing the smoothing part, the independent components' prediction is more accurately performed and also the results are different from the direct prediction on the original time series. The noise in the time series is eliminated, allowing a better prediction of the underlying factors. The model is flexible and allows various smoothing tolerances and different orders in the classical AR-prediction method for each independent component.

For time series which are generated by a linear model – AR-processes, for instance – our algorithm has similar performances with the classical linear time series analysis techniques. Even when such time series are (linearly) mixed, they remain linear, and they can be predicted using standard techniques.

In addition, some of the IC's may also be useful in analyzing the impact of different external phenomena on the foreign exchange rates [2].

In reality, especially in real world signal processing, the model is distorted by delays, noise, and nonlinearities. Some of these can be handled by extensions of the existing ICA algorithms.

6. REFERENCES

[1] S. Amari, A. Cichocki, and H.H. Yang. A new learning algorithm for blind source separation. In *Advances in Neural Information Processing 8*, Cambridge, MA: MIT Press, 757 - 763, 1996.

[2] A. Back and A. Weigend. First application of Independent Component Analysis to extracting structure from stock returns. *Int. J. Neural Systems 8*, 473-484, 1997.

[3] H. B. Barlow. Single units and sensation: a neuron doctrine for perceptual psychology? *Perception*, 1: 371 - 394, 1972.

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

[5] J.-F. Cardoso. Blind signal separation: statistical principles. *Proc. IEEE 86*, 2009 - 2025, 1998.

[6] A. Cichocki and R. Unbehauen. Robust neural networks with on-line learning for blind identification and blind separation of sources. *IEEE Trans. CS I*, 43: 894 - 906, 1996.

[7] P. Comon. Independent Component Analysis - a new concept? *Signal Processing*, 36:287 - 314, 1994.

[8] The FastICA public domain package available at `http://www.cis.hut.fi/projects/ica/fastica/`.

[9] J. Friedman. Exploratory projection pursuit. *J. Am. Stat. Assoc.*, 82: 249 - 266, 1987.

[10] X. Giannakopoulos, J. Karhunen, and E. Oja. An experimental comparison of neural ICA algorithms. *Proc. ICANN'98*, Sept. 2 - 4, 1998, Skövde, Sweden.

[11] X. Giannakopoulos, J. Karhunen, and E. Oja. An experimental comparison of neural algorithms for Independent Component Analysis and Blind Separation. *Int. J. Neural Systems*, 9: 99 - 114, 1999.

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

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

[14] A. Hyvärinen and E. Oja. Independent component analysis by general non-linear Hebbian-like learning rules. *Signal Processing*, 64: 301 - 313, 1998.

[15] A. Hyvärinen. Denoising of sensory data by maximum likelihood estimation of sparse components. *Proc. Int. Conf. on Artificial Neural Networks ICANN'98*, Sept. 2-4, 1998, Skövde, Sweden.

[16] A. Hyvärinen. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Transactions on Neural Networks*, 10(3):626–634, 1999.

[17] C. Jutten and J. Herault. Independent component analysis (INCA) versus independent component analysis. In *Signal Processing IV: Theories and Applications* (J. Lacoume et al, eds.), Elsevier, 643 - 646, 1988.

[18] J. Karhunen, E. Oja, L. Wang, R. Vigario, and J. Joutsensalo. A class of neural networks for Independent Component Analysis. *IEEE Trans. on Neural Networks*, 8: 486 - 504, 1997.

[19] K. Kiviluoto and E. Oja. Independent Component Analysis for parallel financial time series. *Proc. Int. Conf. on Neural Information Processing ICONIP'98*, Oct. 21 - 23, 1998, Kitakyushu, Japan.

[20] S. Malaroiu, K. Kiviluoto, and E. Oja. Time series prediction with Independent Component Analysis. To appear in *Proc. AIT99*.

[21] E. Oja. *Subspace Methods of Pattern Recognition*. Research Studies Press, Letchworth, 1983.

[22] E. Oja. The nonlinear PCA learning rule in independent component analysis. *Neurocomputing*, 17: 25 - 45, 1997.

[23] E. Oja. From neural learning to independent components. *Neurocomputing*, 22: 187 - 200, 1998.

[24] P. Pajunen. Blind source separation using algorithmic information theory. *Neurocomputing*, 22: 35 - 48, 1998.

[25] H. Yang and S. Amari. Adaptive online learning algorithms for blind separation: maximum entropy and minimum mutual information. *Neural Computation*, 9: 1457 - 1482, 1997.

[26] A. S. Weigend, N.A. Gershenfeld. Time Series Prediction. Proceedings of NATO Advanced Research Workshop on Comparative Time Series Analysis. Santa Fe, New Mexico, May 14-17, 1992.