Audio Engineering Society

Convention Paper

Presented at the 110th Convention
2001 May 12–15 Amsterdam, The Netherlands

Computation of Generalized Mutual Information from Multichannel Audio Data

Joshua D. Reiss, Nikolaos Mitianoudis, Mark Sandler
Department of Electronic Engineering, King's College, University of London, Strand, London WC2R 2LS, U.K.

ABSTRACT

The authors present a new method to extract the mutual information for data from any number of channels from either a discrete or continuous system. This generalized mutual information allows for the estimation of the average number of redundant bits in a vector measurement. Thus it provides insight into the information shared between all channels of the data. It may be used as a measure for the success of blind signal separation with multichannel audio. Several multichannel audio signals are separated using various ICA methods and the mutual information of each signal is computed and interpreted. It is also implemented as a contrast function in ICA for a new method of blind signal separation.

INTRODUCTION

Information theoretic methods have a vast amount of applications in audio processing. They are at the core of most compression methods, give theoretical limits for transmission rates and provide measures of loss in analogue to digital conversion. At the heart of information theory is the definition of the mutual information between two signals. It is used in the compression of multichannel data, in the separation of signals, and in Hidden Markov Model based speech recognition systems. Although the mutual information between two signals can be estimated easily under many circumstances, the estimation of the mutual information between many channels for finite data where little is known of the underlying system (e. g., unknown, possibly continuous alphabets) is a challenging problem. Yet this general case may occur any time information theoretic methods are applied to unknown multichannel systems.

The mutual information of two continuous random variables, X and Y with joint density function $f(x,y)$, is given by

$$I = I(X;Y) = \int f(x,y)\log\frac{f(x,y)}{f(x)f(y)}dxdy \qquad (1)$$

where the convention $0\log 0 = 0$ is used.

The logarithm is typically taken to base 2 so that the mutual information is given in units of bits. For a discrete distribution where the two random variables are defined over alphabets, $X \in \mathrm{X}$ and $Y \in \Psi$, this becomes

$$I(X;Y) = \sum_{x\in \mathrm{X}}\sum_{y\in\Psi} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \qquad (2)$$

which is equivalent to (1) in the limit of Χ and Ψ approaching the support sets of X and Y respectively. This provides a measure of the amount of information that a measurement of X contains regarding the measurement of Y, and vice-versa. On the other hand, if X and Y have equal probability mass functions, $p(x) = p(y)$, then the mutual information function assumes its maximum value,

$$I(X;X) = -\sum_{x \in X} p(x) \log p(x) \qquad (3)$$

which is the Shannon entropy of X.

Estimation of the mutual information of two variables is typically performed using a joint histogram. This can be extended to the estimation of the mutual information between n variables from direct estimation of the joint histogram

$$I_n = I(X_1, X_2, ... X_n) = \sum p(x_1, x_2, ... x_n) \log \frac{p(x_1, x_2, ... x_n)}{p(x_1) p(x_2) ... p(x_n)} \qquad (4)$$

However, in practice this has both theoretical and implementational difficulties. For a histogram with N bins, the n dimensional joint histogram would require N^n bins. Thus for large, multichannel data sets, standard histogram estimation of the mutual information would be beyond the memory capacity of many modern computers. If the joint histogram is not kept in memory but instead estimated from knowledge of the binning in the individual histograms, then the number of computations assumes an exponential dependence on the number of channels.

But both these difficulties are of minor importance in comparison to the practical issue of how to estimate a limit in the presence of only finite data. That is, (4) approaches

$$I_n = I(X_1, X_2, ... X_n) = \int f(x_1, ... x_n) \log \frac{f(x_1, ... x_n)}{f(x_1) ... f(x_n)} dx_1, ... dx_n \qquad (5)$$

in the limit of infinitely small box size. But for $X_1, X_2, ... X_n$ defined on infinite alphabets, e. g., $(x_1, ... x_n) \in R^n$, then histogram estimation of the mutual information is dependent on bin size. If the bins are small enough that at most one data point resides in each bin of the joint histogram, then the mutual information is guaranteed to assume its maximum value. For N points

$$I_n = \sum \frac{1}{N} \log \left(\frac{1}{N} \Big/ \left(\frac{1}{N} \right)^n \right) = n - 1 \qquad (6)$$

On the other hand, if the bin size is too large, then the fine scale structure of the data is completely ignored and the resulting uniformity leads to a severe underestimation of the mutual information.

This paper builds on the method of computing mutual information that was suggested by Fraser and Swinney.[1] In that work, they suggested a method of estimating the mutual information between two one-dimensional time series. It was specifically applied to data from a continuous alphabet where the number of data points was a power of two, and later extended to multiple channels.[1-3] We extend that method to the generalized case of an arbitrary number of channels, arbitrary number of data points and an unknown (discrete or continuous) alphabet. In addition, we show how an alternative algorithm greatly reduces memory requirements and increases computational speed.

In the following section, we describe an efficient method of computing I_n. In the Results section, I_n is computed for audio data from various blind signal separation algorithms for 2, 3, and 5 channel signals. It is shown that, for the separation of multichannel signals, I_n is a strong indicator of the effectiveness of signal separation.

METHOD

The data is described as a series of vectors $\overline{X}(1), \overline{X}(2), ..., \overline{X}(N)$ where each vector is n dimensional, $\overline{X}(i) = (X_1(i), X_2(i), ... X_n(i))$, and the number of vectors is a power of two, N=2^K. The vectors are assumed to be sampled uniformly in time. That is, at each time interval Δt, a new vector is sampled, and Δt is a constant. Alternatively, the data may be described as n scalar time series, $\overline{X_1}, \overline{X_2}, ... \overline{X_n}$, where each time series consists of N data points, $\overline{X_j} = (X_j(1), X_j(2), ... X_j(N))$. We will use both of these notations depending on the situation. For the sake of simplicity, we use a slightly different notation for the generalized mutual information than was suggested by Fraser and Swinney. The multichannel mutual information for time series data may be written as

$$I_n = \sum_{x_1 \in \overline{X_1}} \sum_{x_2 \in \overline{X_2}} ... \sum_{x_n \in \overline{X_n}} p(x_1, x_2, ... x_n) \log \frac{p(x_1, x_2, ... x_n)}{p(x_1) p(x_2) ... p(x_n)} \qquad (7)$$

so that I_2 is the mutual information as typically defined.

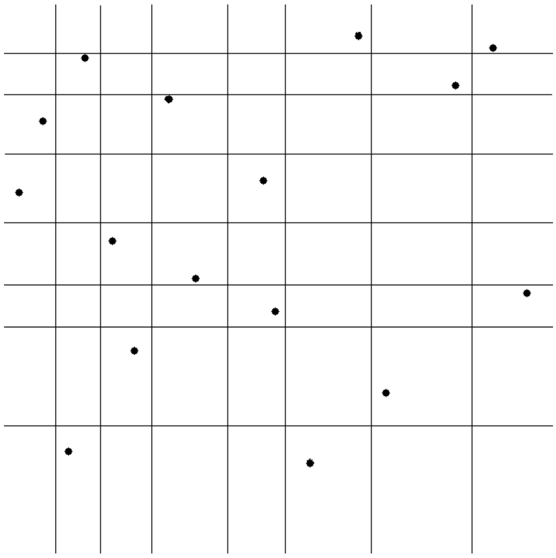

Figure 1. An example grid used to compute the mutual information of two time series. The cuts are made such that a point may occupy each row or column with equal probability.

Although all the audio data analyzed in the Results section come from 16 bit wavefiles, we would like to compute the mutual information for data of arbitrary precision. A change of variables is first performed on the data, so that each time series $\overline{X_j} = (X_j(1), X_j(2), ... X_j(N))$ is transformed into integers, $\overline{Y_j} = (Y_j(1), Y_j(2), ... Y_j(N))$ such that the integer values fill the range 0 to N-1 and $X_j(i) < X_j(k)$ implies $Y_j(i) < Y_j(k)$. The precision of the data is thus dictated by the number of data points, i.e., 2^8 data points give data with 8 bit precision. This has the effect of gridding each dimension into equiprobable partitions.

Fig. 1 depicts the partitions for the case of 16 2-dimensional data points. The equiprobable binning has the added benefit that $p(x_1) = p(x_2) ... = p(x_n)$. Notice that theoretically, this should yield the same estimate for the mutual information in the limit of infinite data since the mutual information is constant with respect to invariant transformations. Issues arise here, however, since the estimation method used may not have the same invariance. Nevertheless, this method is still a reasonable approximation of the mutual information for the transformed data.

In [1] a recursive approach was taken to the estimation of the mutual information. This can be extended to the n dimensional case by creating a 2^n – ary tree from the data $\overline{Y_1}, \overline{Y_2}, ..., \overline{Y_N}$. At each level of the tree a vector is sorted into one of the 2^n branches, depending on the value of each vector coordinate. At each node in the tree, the number of elements in or below that node is stored. Such a tree is known as a quadtree and its creation and searching are discussed extensively in computer science literature.[4-6]

The mutual information is computed through a traversal of the tree. Recall that N is a power of two (if not, then data points are removed), $N = 2^K$. The levels of the tree define successive partitions of the n-dimensional space. For the m^{th} level of the tree, the space is partitioned into 2^{mn} hypercubes, $R_m(0), R_m(1), ...R_m(2^{nm}-1)$ such that the hypercube $R_m(j)$ may be partitioned into $R_{m+1}(2^n j), R_{m+1}(2^n j+1), ...R_{m+1}(2^n j + 2^n - 1)$. Each hypercube has an associated probability $P_m(j)$, which is estimated from the relative frequency of that hypercube, $N_m(j)/N$. Thus the n-dimensional mutual information may be estimated for any level m of the tree.

$$i_m = \sum_{j=0}^{2^{nm}-1} P_m(j) \log \frac{P_m(j)}{P_{1,m}(j) P_{2,m}(j) ... P_{n,m}(j)} \tag{8}$$

where $P_{i,m}(j)$ is the probability of finding the i^{th} coordinate of a vector to reside in the same partition along the i^{th} direction as $R_m(j)$. Due to the equiprobable nature of the partitions, $P_{i,m}(j) = 2^{-m}$ for all i and j. Hence

$$i_m = mn + \sum_{j=0}^{2^{nm}-1} P_m(j) \log P_m(j) \tag{9}$$

Note that the contribution to i_m of $R_m(j)$ is $mnP_m(j) + P_m(j)\log P_m(j)$ and the contribution to i_{m+1} of $R_m(j)$ is $(m+1)nP_m(j) + \sum_{k=2^n j}^{2^n(j+1)-1} P_{m+1}(k) \log P_{m+1}(k)$. So in going from i_m to i_{m+1} a $nP_m(j)$ term is added and the $P_m(j)\log P_m(j)$ term is replaced by $\sum_{k=2^n j}^{2^n(j+1)-1} P_{m+1}(k) \log P_{m+1}(k)$. If the m^{th} level has no substructure, then $P_{m+1}(2^n j) =$ $P_{m+1}(2^n j+1) = ...$ $P_{m+1}(2^n(j+1)-1) = P_m(j)/2^n$. So, without substructure, the contribution to i_{m+1} of $R_m(j)$ would be the same as the contribution to i_m of $R_m(j)$. Therefore $I_n = G_0(0)$ where,

$$G_m(j) = P_m(j)\log P_m(j) \text{ , if there is no substructure and} \tag{10}$$

$$G_m(j) = nP_m(j) + \sum_{k=2^n j}^{2^n(j+1)-1} G_{m+1}(k) \text{ , if there is substructure.} \tag{11}$$

Fraser and Swinney suggest a χ-squared test for substructure. However, this is a source of error and is unnecessary. Instead, we choose to stop the calculation when no further subdivision of $R_m(j)$ is possible, and hence we are guaranteed no substructure. This is the case if $N_m(j) \le 1$, since a cube containing 1 point will always be partitioned into 2^n-1 cubes containing no points and a cube containing 1 point. This partitioning is clearly an artifact of the finite number of data points and not of any structure in the data.

So using this cutoff as a test for substructure, we have $I_n = \frac{nF_0(0)}{N} - \log N$ where,

$$F_m(j) = 0 \text{ , if } N(R_m(j)) < 2 \tag{12}$$

$$F_m(j) = N_m(j) + \sum_{k=2^n j}^{2^n(j+1)-1} F_{m+1}(k) \text{ , if } N(R_m(j)) \ge 2 \text{ .} \tag{13}$$

Each node $R_m(j)$ in the tree contains $N_m(j)$ points and pointers to $R_{m+1}(2^n j)$, $R_{m+1}(2^n j+1)$, $...R_m(2^n j + 2^n - 1)$. The tree is traversed from left to right using equations (12) and (13) to keep a running sum of the mutual information.

However, such a search tree based approach is problematic. First, the memory required to store the tree is on the order of $2^n N$, due to the fact that null pointers must be created at each node. In addition to the memory problems, this implies that a great deal of computational time must be spent dynamically allocating (and deallocating) the tree. This is unnecessary since the tree is intended only to be searched once. A far quicker implementation may be performed if one notices a natural ordering for multidimensional data.

For 8 points of two channel data, the sorted scaled data may be ordered in the following manner.

(0,0) -> 00 00 00
(3,2) -> 00 01 10
(2,1) -> 00 10 01
(1,6) -> 01 01 10
(5,3) -> 10 01 11
(4,4) -> 11 00 00
(6,5) -> 11 10 01
(7,7) -> 11 11 11

This is achieved using a quicksort based on the following comparison scheme. For vectors $\overline{Y}(i)$ and $\overline{Y}(j)$, one determines whether $\overline{Y}(i) < \overline{Y}(j)$, $\overline{Y}(i) > \overline{Y}(j)$, or $\overline{Y}(i) = \overline{Y}(j)$ by making the following comparisons.

If the first bit in the binary representation of $Y_1(i)$ is less than the first bit in the binary representation of $Y_1(j)$, then $\overline{Y}(i) < \overline{Y}(j)$. Else if the first bit in the binary representation of $Y_1(i)$ is greater than the first bit in the binary representation of $Y_1(j)$, then $\overline{Y}(i) > \overline{Y}(j)$. If these first bits are equal, then we make the same comparison on the first bits of $Y_2(i)$ and $Y_2(j)$, and so on up to the n^{th} components of the vectors. If all of these are equal, then we compare the second bits of $Y_1(i)$ and $Y_1(j)$, and then, if necessary the second bits of $Y_2(i)$ and $Y_2(j)$, and so on. Again, if these are all equal we move on to the third bits of the binary representation. This procedure is completed until all bits of the binary representations of $\overline{Y}(i)$ and $\overline{Y}(j)$ have been compared. Thus the comparison of two vectors requires at most $n \cdot K$ binary comparisons.

Computation of the mutual information can then be achieved through processing the sorted data using equations (12) and (13) to keep a running sum of the mutual information. For each level of the binary representation a tally is kept of the number of points within the current box. Two successive vectors in the sorted data are compared. When the two vectors differ on a given level, we know that we have moved to a new box on that and all lower levels, and so the number of points in each of these boxes is added to $F_0(0)$.

The minimum value of the information occurs when $\overline{X_1}, \overline{X_2}, ...\overline{X_n}$ are completely independent. In which case,

$p(x_1, x_2, \ldots x_n) = p(x_1)p(x_2)\ldots p(x_n)$ and $I_n = 0$. The maximum value of the mutual information occurs when $\overline{X_1} = \overline{X_2} = \ldots \overline{X_n}$. In which case $p(x_1, x_2, \ldots x_n) = p(x_1) = p(x_2) = \ldots = p(x_n)$ and

$$I_n = \sum_{j=0}^{2^{nK}-1} P_K(j) \log \frac{P_K(j)}{P_K(j)^n} = (n-1)K \qquad (14)$$

This gives an invariant measure for a data set. $I_n/(n-1)K$ represents the probability of a bit in a measurement being redundant, given the total size of the data set.

RESULTS

Blind signal separation (BSS)[7-9] is the recovery of unobserved independent signals or sources from several observed sources. The classic example of such a problem is when *m* microphones are placed in a room and used to record *n* people speaking simultaneously. BSS then involves the recovery of the *m* voices from the *n* recordings. The general case is depicted in Figure 2.

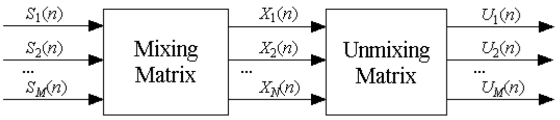

Figure 2. Blind source separation involves the determination of *M* sources when only *N* mixed signals are known. This is done using an unmixing matrix that attempts to reverse the operation of the unknown mixing matrix.

Mutual information is an appropriate statistical measure for the success of a BSS algorithm. If two sources are truly independent, then no information should be shared between them. Highly mixed sources would share considerable information, and identical sources would share the maximum allowable information (the entropy of the source $H(X)$). The same is true of the mutual information as applied to *n* sources, except that the maximum allowable information shared becomes $(n\text{-}1)H(X)$.

Real life microphone recordings were not used because in such a case, the original source signals would not be known. In order to definitively measure the success of a blind source separation algorithm, the source signals must be compared against the output of the separation method. Therefore, microphone recordings were synthetically simulated from prerecorded sources. The microphone recordings were modeled as linearly mixed combinations of the original sources. This is an oversimplification of a real life recording but still suitable for a simple test of the analysis method. A more sophisticated approach would take into consideration reflections and delayed versions of the source signals produced due to the room acoustics, i.e convolved mixtures. For simplicity, the number of original sources and sensors (microphones) is equal. Using the notation given in the Method Section, the mixing process is described by equation (15).

$$\begin{bmatrix} X_1(i) \\ X_2(i) \\ \ldots \\ X_n(i) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & \ldots & \ldots & a_{nn} \end{bmatrix} \begin{bmatrix} S_1(i) \\ S_2(i) \\ \ldots \\ S_n(i) \end{bmatrix} \Rightarrow \overline{X}(i) = A\overline{S}(i) \qquad (15)$$

The BSS problem is to calculate an *unmixing matrix W*, using the observation signals $\overline{X}(1), \overline{X}(2), \ldots \overline{X}(N)$, so that the product $W\overline{X}(i)$ could retrieve the original signal $\overline{S}(i)$, $\overline{U}(i) = W\overline{X}(i) \approx \overline{S}(i)$.

If $W=A^{-1}$, then the original signal can be retrieved exactly. However, the original signal is unknown, and real world mixing of audio signals is convolved (acoustic reflection), somewhat nonstationary (changing room dynamics) and noisy (other signals modelled as noise). Thus W is an approximation to A^{-1}.

In our tests, we have performed BSS using Independent Component Analysis (ICA)[7, 8, 10]. ICA is a group of recently developed linear transformation methods that aim to minimize the statistical dependence of the observed signals. Each of these methods employs a different metric of measuring statistical dependence. One fast and robust method is the FastICA technique,[10] which employs *negentropy* (a normalized version of the differential entropy), as a measurement for the statistical dependence of the observed signals.

Case 1: 3 Channels

Three independent audio signals were linearly mixed and then separated using the FastICA technique. Each monophonic signal consisted of 2^{15} 16-bit data points sampled at 8kHz. The first two signals represented different pieces of music and the third signal was pure Gaussian noise of zero mean and unit variance. Comparison of $I(\overline{X_1}, \overline{X_2}, \ldots \overline{X_n})$, $I(\overline{S_1}, \overline{S_2}, \ldots \overline{S_n})$, and $I(\overline{U_1}, \overline{U_2}, \ldots \overline{U_n})$ should yield insight into the success of the separation.

Time Series	Mutual Information
All 3 input sources	1.251
All 3 mixed signals	5.976
All 3 output signals	1.256
Input Signals 1 and 2	0.906
Input Signals 1 and 3	0.912
Input Signals 2 and 3	0.903
Mixed Signals 1 and 2	1.657
Mixed Signals 1 and 3	1.098
Mixed Signals 2 and 3	2.065
Output Signals 1 and 2	0.929
Output Signals 1 and 3	0.915
Output Signals 2 and 3	0.925

Table 1. Measurements of mutual information from 3 channel audio. Independent sources were linearly mixed and then separated using independent component analysis. Differences in the mutual information between the mixed and unmixed (input and output) signals are compared. The three dimensional mutual information (I_3) used in the first three computations is a better measure of independence than the traditional mutual information (I_2) used for comparing 2 signals.

The results of computation of the mutual information are shown in Table 1. Measurements of I_3 for the input sources and output signals differ by less than 0.5%, thus indicating that the attempt to separate the signals was highly successful. I_3 is much larger for the mixed signals than for either the input signals (4.777 times) or the output signals (4.758 times), and may serve as a measure of the amount of mixing. Although these effects are also seen using I_2, they are much less obvious. I_2 for the mixed signals is only up to 2.065 times as large as I_2 for the input signals and 2.257 times I_2 for the output signals. This indicates that, for 3 signals, I_3 may be an appropriate tool to use in blind signal separation.

Case 2: 5 channels

We chose a particularly difficult case for five channels. Here, there are five male speakers, each saying the same phrase. Each recording was monophonic, and consisted of 2^{14} 16 bit data points sampled at 8 KHz (total duration of 2.048 seconds). The 5 channels were then mixed using the matrix

$$A = \begin{bmatrix} -0.5945 & -0.9695 & -0.1627 & 0.6762 & 0.0056 \\ -0.6026 & 0.4936 & 0.6924 & -0.9607 & 0.4189 \\ 0.2076 & -0.1098 & 0.0503 & 0.3626 & -0.1422 \\ -0.4556 & 0.8636 & -0.5947 & -0.2410 & -0.3908 \\ -0.6024 & -0.0680 & 0.3443 & 0.6636 & -0.6207 \end{bmatrix} \qquad (16)$$

Three different techniques for ICA were used: FastICA,[10] FixedPointICA,[7] and Maximum Likelihood estimates.[11] Table 2 indicates the results of generalized mutual information estimates for each case. All 3 methods do an excellent job of separating the signals. However, each method results in a slightly higher mutual information than in the original source signals. This indicates that the separation was not perfect. [7-9]

Time Series	Mutual Information
All 5 input sources	2.279
All 5 mixed signals	7.239
Separated (FICA)	2.534
Separated (FixedICA)	2.411
Separated (Maximum LikelihoodICA)	2.484

Table 2. Measurements of mutual information from 5 channel audio. Independent sources were linearly mixed and then separated using independent component analysis. Three different implementations of ICA were compared. Each method is very successful at separating the signals, although each method gives slightly more shared information than in the original signal.

Mutual Information as a contrast function

The mutual information has been used as a contrast function in ICA in many previous works. However, these implementations make assumptions about the prior distribution of the sources,[8, 9] or use approximations to the mutual information that may be circumspect.[7] We attempted to use the mutual information as calculated above as a contrast function.

2^{10} points were mixed using the matrix

$$A = \begin{bmatrix} 0.5 & 0.5 \\ 0.3 & 0.7 \end{bmatrix} \qquad (17)$$

The indices of the unmixing matrix were varied in an attempt to minimize the mutual information of the resulting unmixing matrix *A*. Using the mutual information as the contrast function to be minimised, the unmixing is fairly successful

$$WA = \begin{bmatrix} 1.087 & -0.001 \\ 0.000 & 0.762 \end{bmatrix} \qquad (18)$$

This is compared with ICA using the maximum likelihood method, which yields

$$WA = \begin{bmatrix} 0.290 & 0.003 \\ 0.001 & 1.072 \end{bmatrix} \qquad (19)$$

The mutual information actually acts as a better discriminating statistic than the maximum likelihood. This agrees with the view that mutual information is a good candidate as a contrast function.[12]

CONCLUSION

It has been demonstrated that the mutual information between any number of time series can be estimated using a quick and efficient routine. It is an effective method of determining the redundancy in vector measurements and hence can be used as a measure of the success of blind signal separation algorithms. It can also be implemented as a contrast function for use in implementing new ICA methods.

REFERENCES

[1] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," *Phys. Rev. A*, vol. 33, pp. 1134-1140, 1986.

[2] A. M. Fraser, "Reconstructing Attractors from Scalar Time Series: A Comparison of Singular System and Redundancy Criteria," *Physica D*, vol. 34, pp. 391-404, 1989.

[3] A. M. Fraser, "Information and entropy in strange attractors," *IEEE Trans. Information Theory*, vol. 35, pp. 245-262, 1989.

[4] F. P. Preparata and M. I. Shamos, *Computational geometry: An introduction*. New York: Springer-Verlag, 1985.

[5] H. Samet, *The design and analysis of spatial data structures*: Addison-Wesley, 1989.

[6] H. Hinterberger, K. C. Sevcik, and J. Nievergelt, *ACM Trans. On Database Systems*, vol. 9, pp. 38, 1984.

[7] A. Hyvärinen and E. Oja, "Independent Component Analysis: Algorithms and Applications," *Neural Networks*, vol. 13, pp. 411-430, 2000.

[8] P. J. Smaragdis, "Information Theoretic Approaches to Source Separation," in *MIT Media Lab*: MIT, 1997.

[9] J.-F. Cardoso, "Blind signal separation: statistical principles," *Proceedings of the IEEE*, vol. 90, pp. 2009-2026, 1998.

[10] A. Hyvärinen, "Fast and Robust Fixed-Point Algorithms for Independent Component Analysis," *IEEE Transactions on Neural Networks*, vol. 10, pp. 626-634, 1999.

[11] B. A. Pearlmutter and L. C. Parra, "Maximum Likelihood Blind Source Separation: A Context-Sensitive Generalization of ICA," in *Advances in Neural Information Processing Systems 9*: MIT Press, 1997.

[12] P. Comon, "Independent component analysis, a new concept?," *Signal Processing*, vol. 36, pp. 287-314, 1994.