Demixing Jazz-Music: Sparse Coding Neural Gas for the Separation of Noisy Overcomplete Sources

Kai Labusch and Erhardt Barth and Thomas Martinetz
University of Lübeck - Institute for Neuro- and Bioinformatics
Ratzeburger Alle 160 23538 Lübeck -Germany

Abstract: We consider the problem of separating noisy overcomplete sources from linear mixtures, i.e., we observe N mixtures of $M > N$ sparse sources. We show that the "Sparse Coding Neural Gas" (SCNG) algorithm [8, 9] can be employed in order to estimate the mixing matrix. Based on the learned mixing matrix the sources are obtained by orthogonal matching pursuit. Using synthetically generated data, we evaluate the influence of (i) the coherence of the mixing matrix, (ii) the noise level, and (iii) the sparseness of the sources with respect to the performance that can be achieved on the representation level. Our results show that if the coherence of the mixing matrix and the noise level are sufficiently small and the underlying sources are sufficiently sparse, the sources can be estimated from the observed mixtures. In order to apply our method to real-world data, we try to reconstruct each single instrument of a jazz audio signal given only a two-channel recording. Furthermore, we compare our method to the well-known FastICA [4] algorithm and show that in case of sparse sources and presence of additive noise, our method provides a superior estimation of the mixing matrix.

Key words: *blind source separation, self-organizing maps, neural gas, unsupervised learning, sparse coding, matching prusuit*

1. Introduction

Suppose we are given a number of observations $X = (\mathbf{x}_1, \ldots, \mathbf{x}_L)$, $\mathbf{x}_j \in \mathbb{R}^N$ that are a linear mixture of a number of sparse sources $S = (\mathbf{s}_1, \ldots, \mathbf{s}_M)^T = (\mathbf{a}_1, \ldots, \mathbf{a}_L)$, $\mathbf{s}_i \in \mathbb{R}^L$ and $\mathbf{a}_j \in \mathbb{R}^M$:

$$\mathbf{x}_j = C\mathbf{a}_j + \boldsymbol{\epsilon}_j \quad \|\boldsymbol{\epsilon}_j\| \leq \delta \ . \tag{1}$$

Here $C = (\mathbf{c}_1, \ldots, \mathbf{c}_M), \mathbf{c}_j \in \mathbb{R}^N$ denotes the mixing matrix. We require $\|\mathbf{c}_j\| = 1$ without loss of generality. The vector $\mathbf{a}_j = (s_{1,j}, \ldots, s_{M,j})^T$ contains the contribution of the sources $\mathbf{s}_i$ to the mixture $\mathbf{x}_j$. Additionally, a certain amount of additive noise $\boldsymbol{\epsilon}_j$ is present. Is it possible to estimate the sources $\mathbf{s}_i$ only from the mixtures $\mathbf{x}_j$ without knowing the mixing matrix C? In the past, a number of methods have been proposed that can be used to estimate the $\mathbf{s}_i$ and C knowing only the mixtures $\mathbf{x}_j$ assuming $\|\boldsymbol{\epsilon}_j\| = 0$ and $M = N$ [1]. Some methods assume that the sources are statistically independent [4]. More recently the problem of estimating an overcomplete set of sources has been studied that arises if M is larger than the number of observed mixtures N [6, 10, 11, 19]. Fewer approaches have been proposed for source separation under the presence of noise [5]. An overview of the broad field of blind source separation and independent component analysis (ICA) can be found in [7].

Sparse coding considers a generative data model that is closely related to (1). In this data model $\mathbf{x}_j$ is obtained from a sparse linear combination of some unknown dictionary C. There is evidence that sparse coding is a principle employed by biological systems for signal processing [16]; sparse models have been successfully used to mimic properties of simple cells in the primary visual cortex [15]. In [8, 9] we have proposed the "Sparse Coding Neural Gas" (SCNG) algorithm in order to learn sparse overcomplete data representations under the presence of additive noise. Here we show how a slightly modified version of the same algorithm can be employed to tackle the problem of source separation in a noisy overcomplete setting. We do not make assumptions regarding the type of noise, but our method requires that the underlying sources $\mathbf{s}_i$ are sufficiently sparse; in particular, it requires that the $\mathbf{a}_j$ have to be sparse and that the noise level δ as well as the number of sources M is known.

1.1 Source separation and orthogonal matching pursuit

Recently some properties of the orthogonal matching pursuit algorithm (OMP) [17] with respect to the obtained performance on the representation level have been shown [3]. These results provide the theoretical foundation that allows us to apply OMP to the problem of source separation. We here briefly discuss the most important aspects with respect to our work.

Our method does not require that the sources $\mathbf{s}_i$ are independent but it requires that only a few sources contribute to each mixture $\mathbf{x}_j$, i.e., that the $\mathbf{a}_j$ are sparse. However, an important observation is that if the underlying components $\mathbf{s}_i$ are sparse and independent, for a given mixture $\mathbf{x}_j$ the vector $\mathbf{a}_j$ will be sparse, too.

In order to apply the OMP algorithm to problem (1) let us assume that we know the mixing matrix C. Let us further assume that we know the noise level δ. Let $\mathbf{a}_j$ be the vector containing a small number k of non-zero entries such that

$$\mathbf{x}_j = C\mathbf{a}_j + \boldsymbol{\epsilon}_j \quad \|\boldsymbol{\epsilon}_j\| \leq \delta \tag{2}$$

holds for a given observation $\mathbf{x}_j$. OMP provides an estimation $\mathbf{a}_j^{\mathrm{OMP}}$ of $\mathbf{a}_j$ by iteratively constructing $\mathbf{x}_j$ out of the columns of C. Let $C\mathbf{a}_j^{\mathrm{OMP}}$ denote the current approximation of $\mathbf{x}_j$ in OMP and $\boldsymbol{\epsilon}_j$ the residual that still has to be constructed. Let U denote the set of indices of those columns of C that already have been used during OMP. The number of elements in U, i.e., $|U|$, equals the number of OMP iterations that have been performed so far. The columns of C that are indexed by U are denoted by C^U. Initially, $\mathbf{a}_j^{\mathrm{OMP}} = 0$, $\boldsymbol{\epsilon}_j = \mathbf{x}_j$ and $U = \emptyset$. OMP works as follows:

1. Select $\mathbf{c}_{l_{\mathrm{win}}}$ by $\mathbf{c}_{l_{\mathrm{win}}} = \arg\max_{\mathbf{c}_l, l \notin U}(\mathbf{c}_l^T \boldsymbol{\epsilon}_j)$

2. Set $U = U \cup l_{\mathrm{win}}$

3. Solve the optimization problem $\mathbf{a}_j^{\text{OMP}} = \arg\min_{\mathbf{a}} \|\mathbf{x}_j - C^U \mathbf{a}\|_2^2$

4. Obtain current residual $\boldsymbol{\epsilon}_j = \mathbf{x}_j - C\mathbf{a}_j^{\text{OMP}}$

5. Continue with step 1 until $\|\boldsymbol{\epsilon}_j\| \leq \delta$

It can be shown that

$$\|\mathbf{a}_j^{\text{OMP}} - \mathbf{a}_j\| \leq \Lambda_{\text{OMP}}\ \delta \tag{3}$$

holds if the smallest entry in $\mathbf{a}_j$ is sufficiently large and the number of non-zero entries in $\mathbf{a}_j$ is sufficiently small. Let

$$H(C) = \max_{1 \leq i,j \leq M, i \neq j} |\mathbf{c}_i^T \mathbf{c}_j| \tag{4}$$

be the mutual coherence of the mixing matrix C. The smaller $H(C)$, N/M and k are, the smaller Λ_{OMP} becomes and the smaller $\min(\mathbf{a}_j)$ is allowed to be [3]. Since (3) only holds if the smallest entry in $\mathbf{a}_j$ is sufficiently large, OMP has the property of local stability with respect to (3) [3]. Furthermore, it can be shown that under the same conditions $\mathbf{a}_j^{\text{OMP}}$ contains only non-zeros that also appear in $\mathbf{a}_j$ [3]. An even globally stable approximation of $\mathbf{a}_j$ can be obtained by methods such as basis pursuit [3, 2].

1.2 Optimized Orthogonal Matching Pursuit (OOMP)

The "Sparse Coding Neural Gas" algorithm is based on "Optimized Orthogonal Matching Pursuit" (OOMP) which is an improved variant of OMP[18]. In general, the columns of C are not pairwise orthogonal. Hence, the criterion of OMP that selects the column $\mathbf{c}_{l_{\text{win}}}, l_{\text{win}} \notin U$ of C that is added to U is not optimal with respect to the minimization of the residual that is obtained after the column $\mathbf{c}_{l_{\text{win}}}$ has been added. Hence OOMP runs through all columns of C that have not been used so far and selects the one that yields the smallest residual:

1. Select $\mathbf{c}_{l_{\text{win}}}$ such that $\mathbf{c}_{l_{\text{win}}} = \arg\min_{\mathbf{c}_l, l \notin U} \min_{\mathbf{a}} \|\mathbf{x}_j - C^{U \cup l}\mathbf{a}\|$

2. Set $U = U \cup l_{\text{win}}$

3. Solve the optimization problem $\mathbf{a}_j^{\text{OOMP}} = \arg\min_{\mathbf{a}} \|\mathbf{x}_j - C^U \mathbf{a}\|_2^2$

4. Obtain current residual $\boldsymbol{\epsilon}_j = \mathbf{x}_j - C\mathbf{a}_j^{\text{OOMP}}$

5. Continue with step 1 until $\|\boldsymbol{\epsilon}_j\| \leq \delta$

Step (1) involves $M - |U|$ minimization problems. In order to reduce the computational complexity of this step, we employ a temporary matrix R that has been orthogonalized with respect to C^U. R is obtained by removing the projection of the columns of C onto the subspace spanned by C^U from C and setting the norm of the residuals $\mathbf{r}_l$ to one. The residual $\boldsymbol{\epsilon}_j^U$ is obtained by removing the projection of $\mathbf{x}_j$ to the subspace spanned by C^U from $\mathbf{x}_j$. Initially, $R = (\mathbf{r}_1, \ldots, \mathbf{r}_l, \ldots, \mathbf{r}_M) = C$

and $\epsilon_j^U = \mathbf{x}_j$. In each iteration, the algorithm determines the column $\mathbf{r}_l$ of R with $l \notin U$ that has maximum overlap with respect to the current residual ϵ_j^U:

$$l_{\text{win}} = \arg\max_{l, l \notin U} (\mathbf{r}_l^T \epsilon_j^U)^2 \ . \tag{5}$$

Then, in the construction step, the orthogonal projection with respect to $\mathbf{r}_{l_{\text{win}}}$ is removed from the columns of R and ϵ_j^U:

$$\mathbf{r}_l = \mathbf{r}_l - (\mathbf{r}_{l_{\text{win}}}^T \mathbf{r}_l)\mathbf{r}_{l_{\text{win}}}, \tag{6}$$

$$\epsilon_j^U = \epsilon_j^U - (\mathbf{r}_{l_{\text{win}}}^T \epsilon_j^U)\mathbf{r}_{l_{\text{win}}} \ . \tag{7}$$

After the projection has been removed, l_{win} is added to U, i.e., $U = U \cup l_{\text{win}}$. The columns $\mathbf{r}_l$ with $l \notin U$ may be selected in the subsequent iterations of the algorithm. The norm of these columns is set to unit length. If the stopping criterion $\|\epsilon_j^U\| \leq \delta$ has been reached, the final entries of $\mathbf{a}_j^{\text{OOMP}}$ can be obtained by recursively collecting the contribution of each column of C during the construction process, taking into account the normalization of the columns of R in each iteration. The selection criterion (5) ensures that the norm of the residual ϵ_j^U obtained by (7) is minimal. Hence, the OOMP algorithm can provide an approximation of $\mathbf{a}_j$ containing even fewer non-zeros than the approximation provided by OMP.

2. Learning the mixing matrix

We now consider the problem of estimating the mixing matrix $C = (\mathbf{c}_1, \ldots, \mathbf{c}_M)$ from the mixtures $\mathbf{x}_j$ provided that we know the noise level δ and the number of underlying sources M. As a consequence of the sparseness of the underlying sources $\mathbf{s}_i$, we are looking for a mixing matrix C that minimizes the number of non-zero entries of $\mathbf{a}_j^{\text{OOMP}}$, i.e., the number of iteration steps of the OOMP algorithm, given a noise level of δ

$$\min_C \frac{1}{L} \sum_{j=1}^{L} \|\mathbf{a}_j^{\text{OOMP}}\|_0 \quad \text{subject to} \quad \forall j \ : \ \|\mathbf{x}_j - C\mathbf{a}_j^{\text{OOMP}}\| \leq \delta \ . \tag{8}$$

Here $\|\mathbf{a}_j^{\text{OOMP}}\|_0$ denotes the number of non-zero entries in $\mathbf{a}_j^{\text{OOMP}}$. In order to minimize (8), we perform an update of R and C prior to the construction step (6) and (7) in each iteration of the OOMP algorithm. In order to reduce the total number of iterations, this update step minimizes the norm of the residual that is going to be obtained in the current iteration. The norm of the residual becomes small if

$$(\mathbf{r}_{l_{\text{win}}}^T \epsilon_j^U)^2 \tag{9}$$

is large. Hence, we have to consider the optimization problem

$$\max_R \sum_{j=1}^{L} \max_{l, l \notin U} (\mathbf{r}_l^T \epsilon_j^U)^2 \quad \text{subject to} \quad \|\mathbf{r}_l\| = 1 \ . \tag{10}$$

An optimization of (10) can be achieved by using Oja's rule [14], which is

$$\mathbf{r}_{l_{\text{win}}} = \mathbf{r}_{l_{\text{win}}} + \alpha\, y(\boldsymbol{\epsilon}_j^U - y\,\mathbf{r}_{l_{\text{win}}}) \tag{11}$$

with $y = \mathbf{r}_{l_{\text{win}}}^T \boldsymbol{\epsilon}_j^U$ and learning rate α. Instead of updating only the winning column of R, i.e., $\mathbf{r}_{l_{\text{win}}}$, we employ the soft-competitive learning approach of the "Neural Gas" (NG) algorithm [13, 12] in order to update each column of R that might be selected in the next iteration. We determine the sequence

$$-\left(\mathbf{r}_{l_0}^T \boldsymbol{\epsilon}_j^U\right)^2 \leq \cdots \leq -\left(\mathbf{r}_{l_k}^T \boldsymbol{\epsilon}_j^U\right)^2 \leq \cdots \leq -\left(\mathbf{r}_{l_{M-|U|}}^T \boldsymbol{\epsilon}_j^U\right)^2 \quad , \; l_k \notin U\,. \tag{12}$$

Combining Oja's rule with the soft-competitive update of the NG algorithm, we obtain

$$\Delta\mathbf{r}_{l_k} = \Delta\mathbf{c}_{l_k} = \alpha_t e^{-k/\lambda_t} y\left(\boldsymbol{\epsilon}_j^U - y\,\mathbf{r}_{l_k}\right) \tag{13}$$

Here α_t and λ_t are the learning rate resp. neighborhood size at time t:

$$\lambda_t = \lambda_0\left(\lambda_{\text{final}}/\lambda_0\right)^{t/t_{\max}} \tag{14}$$

$$\alpha_t = \alpha_0\left(\alpha_{\text{final}}/\alpha_0\right)^{t/t_{\max}}\;. \tag{15}$$

In [9] we have shown that this update rule corresponds to a stochastic gradient descent with respect to

$$\max_R \sum_{j=1}^{L}\sum_{l=1}^{M} h_{\lambda_t}(k(\mathbf{r}_l, \boldsymbol{\epsilon}_j))(\mathbf{r}_l^T \boldsymbol{\epsilon}_j)^2 \quad \text{subject to} \quad \|\mathbf{r}_l\|_2^2 = 1, \tag{16}$$

with $h_{\lambda_t}(v) = e^{-v/\lambda_t}$. Here $k(\mathbf{r}_l, \boldsymbol{\epsilon}_j)$ denotes the number of columns of the temporary matrix R with $(\mathbf{r}_j^T \boldsymbol{\epsilon}_j)^2 > (\mathbf{r}_l^T \boldsymbol{\epsilon}_j)^2$. Note that for $\lambda_t \to 0$ this optimization problem is equivalent to the optimization problem defined by (10). For $t \to t_{\max}$ one obtains equation (11) as update rule. Note that (13) accumulates the updates of all iterations in the learned mixing matrix C. Due to the orthogonal projection (6) and (7) performed in each iteration, these updates are pairwise orthogonal. Furthermore, note that the columns of the original matrix emerge in random order in the learned mixing matrix. The sign of the columns of the mixing matrix $\mathbf{c}_l$ cannot be determined because multiplying $\mathbf{c}_l$ by -1 corresponds to multiplying $\mathbf{r}_l$ by -1, which does not change (16). The entire SCNG method is shown in Algorithm 1.

3. Experiments

In order to evaluate the performance of the SCNG algorithm with respect to the reconstruction of the underlying sources, we performed a number of experiments on synthetical data. We generated sparse underlying sources $S = (\mathbf{s}_1, \ldots, \mathbf{s}_M)^T = (\mathbf{a}_1, \ldots, \mathbf{a}_L), \mathbf{s}_i \in \mathbb{R}^L, \mathbf{a}_j \in \mathbb{R}^M$. This was done by setting up to k entries of the $\mathbf{a}_j$ to uniformly distributed random values in $[-1, 1]$. For each $\mathbf{a}_j$ the number of non-zero entries was obtained from a uniform distribution in $[0, k]$. We added gaussian distributed noise $\boldsymbol{\epsilon}_j$ such that

$$\mathbf{x}_j = C\mathbf{a}_j + \boldsymbol{\epsilon}_j\;. \tag{17}$$

Algorithm 1 The sparse coding neural gas algorithm for source separation.

initialize $C = (\mathbf{c}_1, \ldots, \mathbf{c}_M)$ using uniform random values
for $t = 0$ to $t_{\max}$ **do**
  select random sample $\mathbf{x}$ out of X
  set $\mathbf{c}_1, \ldots, \mathbf{c}_M$ to unit length
  calculate current size of neighborhood: $\lambda_t = \lambda_0 \left(\lambda_{\text{final}}/\lambda_0\right)^{t/t_{\max}}$
  calculate current learning rate: $\alpha_t = \alpha_0 \left(\alpha_{\text{final}}/\alpha_0\right)^{t/t_{\max}}$
  set $U = \emptyset$, $\boldsymbol{\epsilon}^U = \mathbf{x}$ and $R = (\mathbf{r}_1, \ldots, \mathbf{r}_M) = C = (\mathbf{c}_1, \ldots, \mathbf{c}_M)$
  while $\|\boldsymbol{\epsilon}^U\| > \delta$ **do**
    determine $l_0, \ldots, l_k, \ldots, l_{M-|U|}$ with $l_k \notin U$:

$$-(\mathbf{r}_{l_0}^T \boldsymbol{\epsilon}^U)^2 \leq \cdots \leq -(\mathbf{r}_{l_k}^T \boldsymbol{\epsilon}^U)^2 \leq \cdots \leq -(\mathbf{r}_{l_{M-|U|}}^T \boldsymbol{\epsilon}^U)^2$$

    for $k = 1$ to $M - |U|$ **do**
      with $y = \mathbf{r}_{l_k}^T \boldsymbol{\epsilon}^U$ update $\mathbf{c}_{l_k} = \mathbf{c}_{l_k} + \Delta_{l_k}$ and $\mathbf{r}_{l_k} = \mathbf{r}_{l_k} + \Delta_{l_k}$ with

$$\Delta_{l_k} = \alpha_t e^{-k/\lambda_t} y \left(\boldsymbol{\epsilon}^U - y\, \mathbf{r}_{l_k}\right)$$

      set $\mathbf{r}_{l_k}$ to unit length
    end for
    determine $l_{\text{win}} = \arg\ \max_{l \notin U} (\mathbf{r}_l^T \boldsymbol{\epsilon}^U)^2$
    remove projection to $\mathbf{r}_{l_{\text{win}}}$ from $\boldsymbol{\epsilon}^U$ and R:

$$\begin{aligned} \boldsymbol{\epsilon}^U &= \boldsymbol{\epsilon}^U - (\mathbf{r}_{l_{\text{win}}}^T \boldsymbol{\epsilon}^U)\mathbf{r}_{l_{\text{win}}} \\ \mathbf{r}_l &= \mathbf{r}_l - (\mathbf{r}_{l_{\text{win}}}^T \mathbf{r}_l)\mathbf{r}_{l_{\text{win}}} \quad , l = 1, \ldots, M \end{aligned}$$

    set $U = U \cup l_{\text{win}}$
  end while
end for

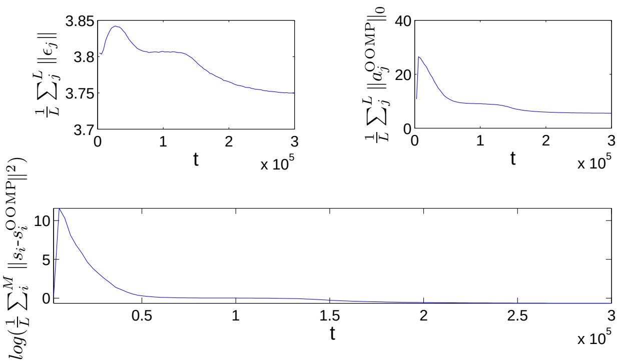

Fig. 1 *The figure shows the convergence of the SCNG algorithm. Top left: The mean norm of the residual $\boldsymbol{\epsilon}_j^U$ of the last iteration of the OOMP algorithm. Middle: Mean number of iterations performed by the OOMP algorithm until $\boldsymbol{\epsilon}_j^U \leq \delta$. Right: Logarithmic plot of the mean squared distance between the estimated sources s_i^{OOMP} and the true sources s_i. We used $M = 100$, $N = 50$, $H(C) = 0.4$, $k = 15$ and $SNR = 5dB$ which corresponds to $\delta = 3.96$. The fraction of non-zero entries in the sources was 7.5%, i.e., the avarage number of non-zero entries of the vectors $\mathbf{a}_j$ was 7.5.*

The noise parameter δ was obtained as

$$\delta = \frac{1}{L}\sum_{j=1}^{L} \|\boldsymbol{\epsilon}_j\| \; . \tag{18}$$

We scaled the S and thereby the $\mathbf{a}_j$ so that $var(CS) = 1$. Then the amplitude of the values in $\boldsymbol{\epsilon}_j$ was chosen such that the desired SNR was obtained. In order to obtain a random mixture matrix $C \in \mathbb{R}^{N\times M}$ with coherence z, we repeatedly chose a matrix from a uniform distribution in $[-1, 1]$ until $\lceil 100H(C)\rceil = \lceil 100z\rceil$. The norm of the columns of the mixture matrix was set to unit length.

In our experiments, we study the error on the representatition level. This means that for each observation $\mathbf{x}_j$, we evaluate the difference between the original contributions of the underlying sources, i.e., $\mathbf{a}_j$ to $\mathbf{x}_j$, and the contributions $\mathbf{a}_j^{\mathrm{OOMP}}$ that were estimated by the OOMP algorithm on the basis of the mixing matrix C that was learned by the SCNG algorithm

$$\frac{1}{L}\sum_{j=1}^{L} \|\mathbf{a}_j - \mathbf{a}_j^{\mathrm{OOMP}}\|_2^2 = \frac{1}{L}\sum_{i=1}^{M} \|\mathbf{s}_i - \mathbf{s}_i^{\mathrm{OOMP}}\|_2^2. \tag{19}$$

Here $S^{\mathrm{OOMP}} = (\mathbf{s}_1^{\mathrm{OOMP}}, \ldots, \mathbf{s}_M^{\mathrm{OOMP}})^T = (\mathbf{a}_1^{\mathrm{OOMP}}, \ldots, \mathbf{a}_L^{\mathrm{OOMP}})$ are the underlying sources obtained from the OOMP algorithm. In order to evaluate (19) we have to assign the entries in $\mathbf{a}_j^{\mathrm{OOMP}}$ to the entries in $\mathbf{a}_j$ which is equivalent to assigning the original sources $\mathbf{s}_i$ to the estimated sources $\mathbf{s}_i^{\mathrm{OOMP}}$. This problem arises due to the random order in which the columns of the original mixing matrix appear in the learned mixing matrix. For the assignment we perform the following procedure:

1. Set $I_{\mathrm{orig}} : \{1, \ldots, M\}$ and $I_{\mathrm{learned}} : \{1, \ldots, M\}$.
2. Find and assign $\mathbf{s}_i$ and $\mathbf{s}_j^{\mathrm{OOMP}}$ with $i \in I_{\mathrm{orig}}, j \in I_{\mathrm{learned}}$ such that

$$\frac{|\mathbf{s}_j^{\mathrm{OOMP}}\mathbf{s}_i^T|}{\|\mathbf{s}_i\|\|\mathbf{s}_j^{\mathrm{OOMP}}\|} \text{ is maximal.}$$

3. Remove i from I_{orig} and j from I_{learned}.
4. If $\mathbf{s}_j^{\mathrm{OOMP}}\mathbf{s}_i^T < 0$ set $\mathbf{s}_j^{\mathrm{OOMP}} = -\mathbf{s}_j^{\mathrm{OOMP}}$.
5. Proceed with (2) until $I_{\mathrm{orig}} = I_{\mathrm{learned}} = \emptyset$.

For all experiments we used $L = 20000$ and $\alpha_0 = 0.1$, $\alpha_{\mathrm{final}} = 0.0001$ for the learning rate as well as $\lambda_0 = M/2$ and $\lambda_{\mathrm{final}} = 10^{-7}$ for the neighborhood size. We repeated all experiments 10 times and report the mean result over the 10 runs. The number of learning iterations of the SCNG algorithm was set to $t_{\mathrm{max}} = 15 * 20000$.

In our first experiment, we evaluated the convergence of the SCNG algorithm over time in case of $N = 50$ observations of $M = 100$ underlying sources with up to $k = 15$ non-zero entries. The result is shown in Figure 1. We set $SNR = 5dB$, which corresponds to setting $\delta = 3.96$. The norm of the residual of the final iteration $\boldsymbol{\epsilon}_j^U$ of the OOMP algorithm converges to a value close to δ. The number of iterations of the OOMP algorithm is reduced over time and converges to a value close to the fraction of non-zero entries in the entire sources, which is 7.5%. Due to $M = 100$ the fraction of non-zero entries is equal to the average number of non-zero entries of the vectors $\mathbf{a}_j$. At the same time also the error on the representation level is minimized. The 5 underlying sources that were estimated best as well as one of the mixtures from which they were obtained are shown in Figure 2.

In the next experiment, we set $N = 20$, $M = 40$ and $SNR = 10dB$ and varied the coherence $H(C)$ of the mixing matrix as well as the sparseness k of the underlying components. The result is shown in Figure 4. The sparser the sources are and the smaller the coherence of the mixing matrix is, the better the obtained performance is. Then, we fixed $H(C) = 0.6$,$SNR = 20dB$ and $k = 5$ and varied the overcompleteness by setting $M = 20, \ldots, 80$. From Figure 4 it can be seen that only slightly varying performance is obtained though the overcompleteness strongly increases. Furthermore we varied N from 10 to 50 and set $M = 2N$ as well as $k = \lceil N/10 \rceil$. Figure 4 shows that almost the same performance is obtained, i.e., the obtained performance does not depend on the number of sources and observations if the fraction N/M as well as the sparseness of the sources is constant.

The results that are shown in Figure 5 were obtained by setting $N = 20$, $M = 40$ and $H(C) = 0.6$ and varying the noise level as well as the sparseness of the sources. As expected, the more noise is present and the less sparse the sources are, the lower the obtained performance is. Finally, we set $k = 5$ and studied the obtained reconstruction performance depending on the coherence of the mixing matrix and the noise level. The result is also shown in Figure 5. It can be seen that in our experimental setting the noise level has an strong impact on the performance. The influence of the noise cannot be compensated by the coherence of the mixing matrix.

4. An application to real world data: Demixing Jazz music

So far, we only have considered synthetical data in our experiments. In order to show that the SCNG algorithm can also be applied successfully to more realistic data, we here apply the algorithm to audio data in a difficult setting.

Let us assume that we are given a recording of a piece of Jazz-music. The given piece is arranged for five instruments, vibraphone, piano, trumpet, percusion and bass. It is recorded with two microphones. Unfortunately, some additive noise is also present during the recording. Now, we want to reconstruct the part of each single instrument given only the recording of the microphones.

In order to implement the setting described above, we took a Jazz midi-file and extracted the five instruments from that file. The obtained five audio tracks each containing a single instrument were used as ground truth. We obtained the simulated two-channel recording by applying a 2×5 mixing matrix to the five audio tracks and adding some gaussian noise to the mixture. Figure 6 shows the two recorded channels that were obtained. Then, we applied the SCNG algorithm to the recordings in order to estimate the mixing matrix.

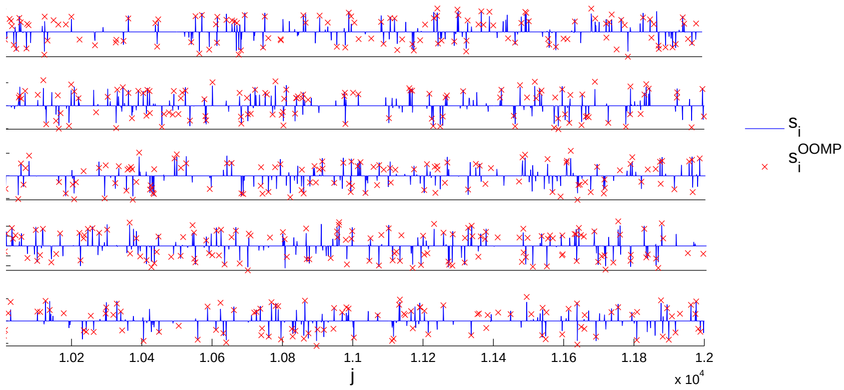

Fig. 2 *The figure shows the* 5 *out of* $M = 100$ *underlying sources* $\mathbf{s}_i$ *that were reconstructed best from the* $N = 50$ *mixtures observed. One of the mixtures is shown in Figure 3 . The solid line depicts* $\mathbf{s}_i$ *whereas the crosses depict the estimated sources* $\mathbf{s}_i^{\mathrm{OOMP}}$ *that were obtained by applying the OOMP to the mixtures using the mixing matrix that was learned by the SCNG algorithm. The coherence of the mixture matrix was set to* $H(C) = 0.4$*. The data was generated using* $k = 15$, $SNR = 5dB$*. We used* $\delta = 3.96$*.*

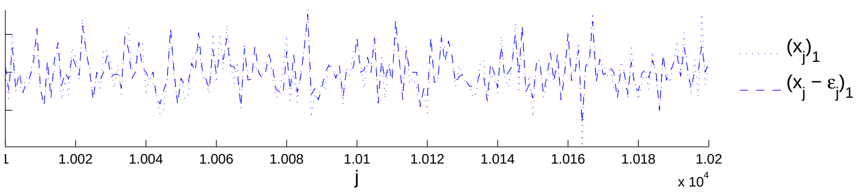

Fig. 3 *The figure shows one out of* 50 *mixtures from which the estimated sources that are shown in Figure 2 were obtained. The dashed line depicts* $(\mathbf{x}_j - \boldsymbol{\epsilon}_j)_1$ *whereas the dotted line depicts* $(\mathbf{x}_j)_1$.

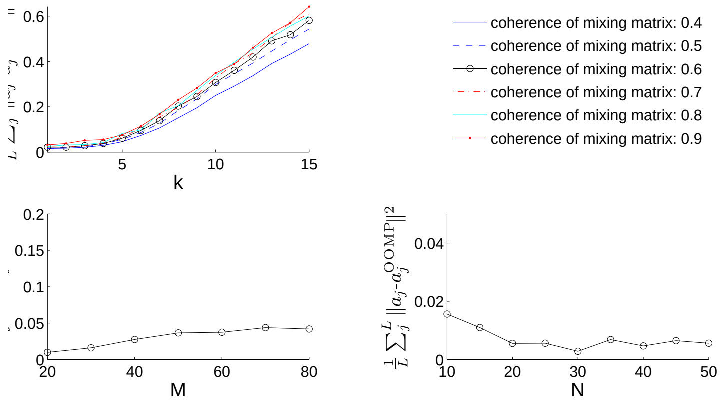

Fig. 4 *Top left: The influence of decreasing sparseness and increasing coherence of the mixing matrix with respect to the reconstruction error is shown. We used* $N = 20$*,* $M = 40$ *and* $SNR = 10dB$*. Bottom left: The obtained reconstruction error using* $M = 20, \ldots, 80$*,*$SNR = 20dB$ *and* $k = 5$*. Bottom right: The obtained reconstruction error for* $N = 10, \ldots, 50$ *with* $M = 2N$*,*$k = \lceil N/10 \rceil$ *and* $SNR = 20dB$*.*

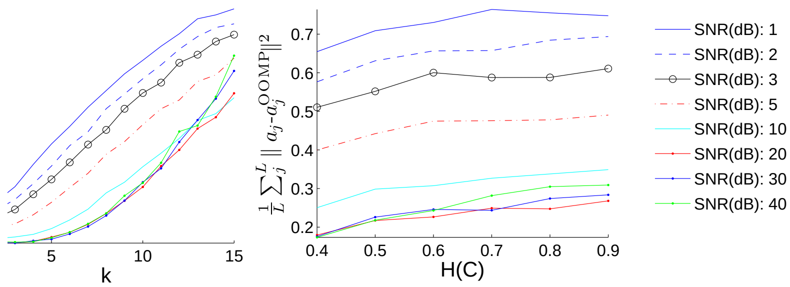

Fig. 5 *Left: We used* $N = 20, M = 40$*. The coherence of the mixing matrix was set to* 0.6*. The reconstruction performance depending on the noise level as well as on the sparseness is shown. Right: The sparseness parameter* k *was set to* 5*. The obtained reconstruction error depending on the noise level and the coherence of the mixing matrix is shown.*

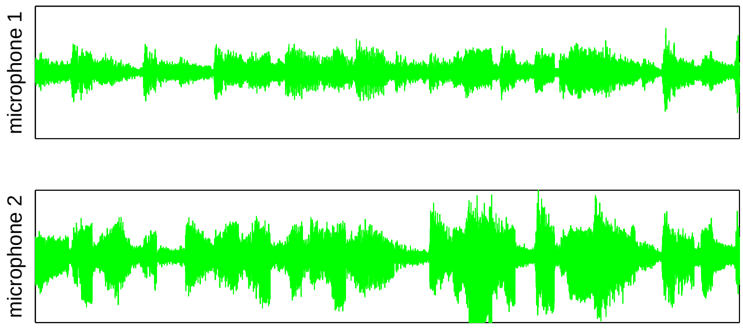

Fig. 6 *The figure shows the two recorded channels that have been obtained by applying a* 2×5 *mixing matrix to the tracks of the instruments that are shown in Figure 7. Gaussian noise has been added to the recorded channels. The coherence of the mixing matrix was set to* 0.5*. The SNR was set to 10 dB.*

In the same way as already described before, we then used the OOMP algorithm in order to estimate the original audio track of each instrument from the recordings. Figure 7 shows the original audio tracks and the estimatated track for each instrument. It can be seen that one can clearly assign estimated and original part of each instrument. The auditory impression of the estimated parts is quite noisy but one can clearly assign each part to an instrument. However, periods of silence are estimated rather badly as can be seen for example from the percussion track. The result in this very difficult setting is promising and it might be improved by employing computationally more demanding methods such as basis pursuit using the mixing matrix provided by the SCNG method in order to reconstruct the tracks of the instruments.

4.1 Comparison to FastICA

Finally, we want to compare our method to another well-known approach that also has been used to perform blind source separation, the FastICA algorithm [4]. The standard FastICA algorithm is not applicable in case of an overcomplete setting. Therefore, in our experiment, there are as many mixtures as sources, i.e., the mixing matrix is invertible. We want to study what the impact of the additive noise is on the reconstruction performance of both methods with respect to the mixing matrix.

We randomly chose a mixing matrix $C^{\mathrm{orig}} \in \mathbb{R}^{20\times 20}$ with mutual coherence 0.5. As described before, we generated sparse sources $\mathbf{a}_j \in \mathbb{R}^{20}$, where the maximum number of non-zero entries of the sources $\mathbf{a}_j$ was set to 5. We applied the mixing matrix to the sources in order to obtain the mixtures $\mathbf{x}_j \in \mathbb{R}^{20}$. We employed the SCNG algorithm and the FastICA algorithm in order to obtain the estimated mixing matrices C^{SCNG} and C^{FICA}. We then compared the estimated mixing matrix to the original mixing matrix by computing the maximum overlap between each column of the original mixing matrix and the learned mixing matrix. For

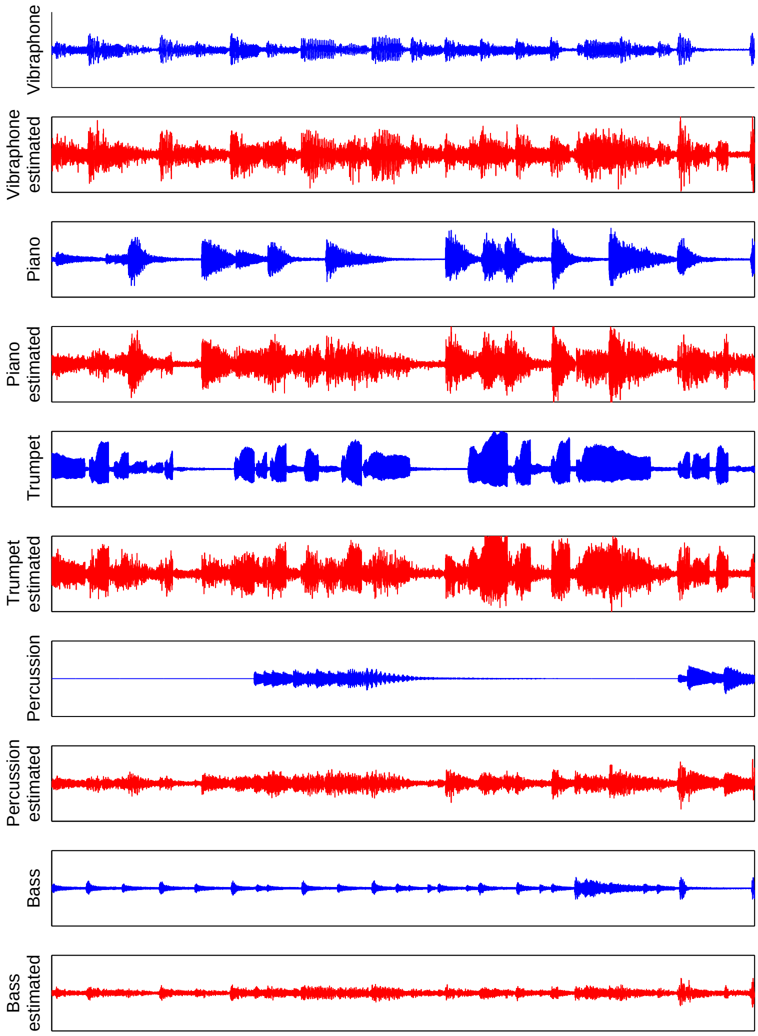

Fig. 7 *The figure shows the original audio track of each instrument as well as the estimated audio track that has been obtained by applying the OOMP algorithm to the two-channel recording that is shown in Figure 6 using the mixing matrix that was learned by the SCNG algorithm.*

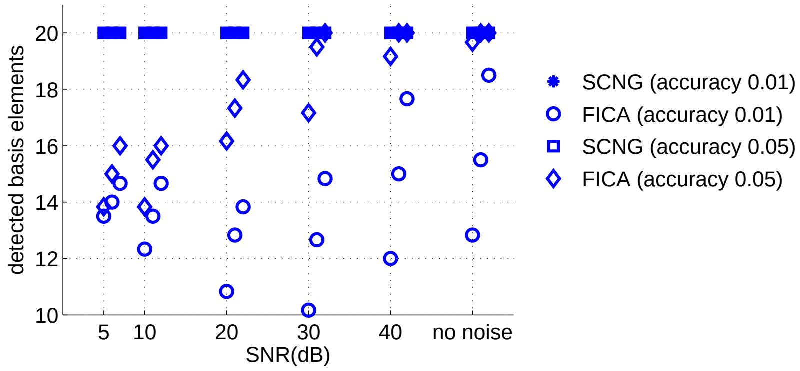

Fig. 8 *The figure shows the reconstruction performance of SCNG and FastICA on synthetically generated data, i.e., the number of successfully learned columns of the mixing matrix up to an accuracy of* 0.01 *and* 0.05 *respectively. We set* $M = 20$*,* $N = 20$ *and* $k = 5$*. The coherence of the mixing matrix was set to* 0.5*. We varied the SNR. In particular in case of the presence of strong noise SCNG outperforms FastICA.*

instance, in case of the solution provided by the SCNG algorithm, we computed

$$\max_j \left(1 - |\mathbf{c}_i^{\text{orig}} \mathbf{c}_j^{\text{SCNG}}|\right) . \tag{20}$$

Whenever (20) was smaller than some accuracy, we counted this as a success.

We repeated this experiment 9 times with varying SNR as well as zero noise. For each noise level we sort the 9 trials according to the number of successfully learned columns of the mixing matrix and order them in groups of 3 experiments. Figure 8 shows the mean number of successfully detected columns for each of the 3 groups for each noise level. We performed the experiment twice using a required accuracy of 0.01 and 0.05. From Figure 8, it can be seen that the SCNG algorithm is not sensitive to the additive noise whereas the performance of the FastICA method degrades with increasing noise level.

In the next experiment, we set the noise level to 10 dB and varied the number of non-zero entries of the sources $\mathbf{a}_j$. The results are shown in Figure 9. Though for $k = 9$ the performance of the SCNG algorithm degrades, it outperforms the FastICA method on this noisy data.

In the last experiment, we used the audio data described in section 4. For the audio data a randomly chosen mixing matrix $C^{\text{orig}} \in \mathbb{R}^{5\times 5}$ with coherence 0.5 was used. The results on the audio data are shown in Figure 10. It can be seen that for accuracy 0.01 FastICA outperforms the SCNG method. The opposite result is obtained for accuracy 0.05. The audio data are less sparse than the data used in the artifical setting, which might cause the reduced performance of the SCNG algorithm.

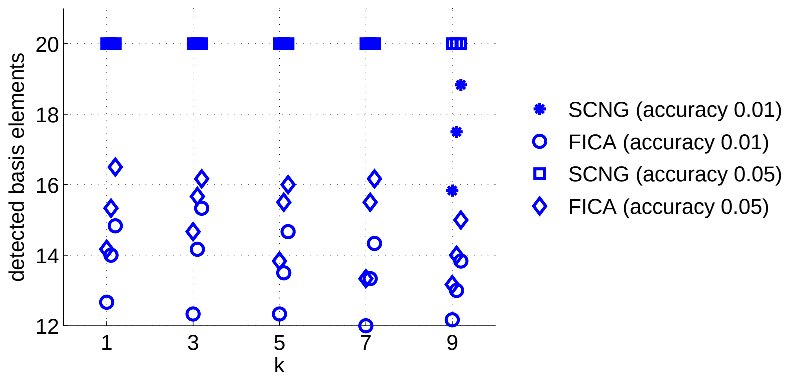

Fig. 9 *The figure shows the reconstruction performance of SCNG and FastICA on synthetically generated data, i.e., the number of successfully learned columns of the mixing matrix up to an accuracy of* 0.01 *and* 0.05*, respectively. We set* $M = 20$, $N = 20$ *and* $SNR = 10dB$*. The coherence of the mixing matrix was set to* 0.5*. We varied the parameter* k *which controls the sparseness of the underlying sources. The larger* k *is, the less sparse the sources are. Again SCNG outperforms FastICA. For large* k *the performance of SCNG degrades.*

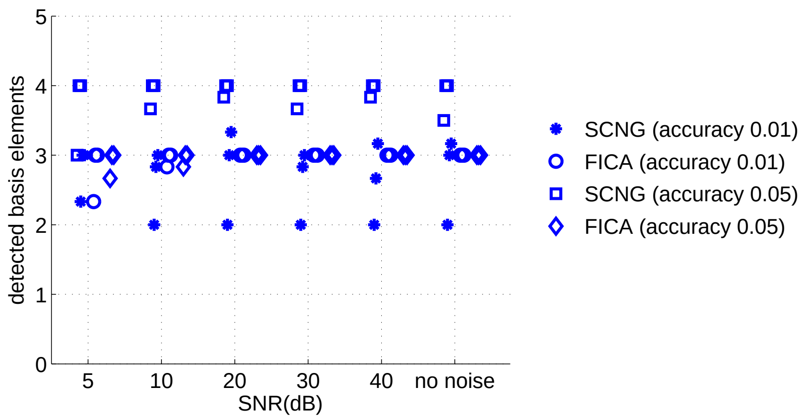

Fig. 10 *The figure shows the reconstruction performance of SCNG and FastICA on the audio data that is described in section 4., i.e., the number of successfully learned columns of the mixing matrix up to an accuracy of* 0.01 *and* 0.05*, respectively. We set* $M = 5$, $N = 5$*. The coherence of the mixing matrix was set to* 0.5*. We varied the SNR. For accuracy* 0.01 *FastICA outperforms SCNG. For accuracy* 0.05 *one obtains the opposite result. In case of strong noise (*$SNR = 5dB$*) SCNG performs better than FastICA.*

5. Conclusion

We introduced the SCNG algorithm in order to tackle the difficult problem of estimating the underlying sources of a linear mixture in a noisy overcomplete setting. Our model does not make assumptions regarding the distribution of the sources or the distribution of the noise. However, the method requires that the sources are sparse and that the noise level as well as the number of the underlying sources are known or can be estimated.

Based on the mixing matrix that was learned by the SCNG algorithm, we evaluated the performance on the representation level by employing the OOMP algorithm in order to obtain the sources from the observations. We analyzed the performance with respect to the reconstruction of the original sources that can be achieved. We studied the influence of the coherence of the mixing matrix, the noise level and the sparseness of the underlying sources. If the sources are sufficiently sparse and the coherence of the mixing matrix and the noise level are sufficiently small, the SCNG algorithm is able to learn the mixing matrix and the sources can be reconstructed. We also evaluated the influence of the overcompleteness with respect to the obtained performance. Our results show that sufficiently sparse sources can be reconstructed even in highly overcomplete settings.

We also have shown that the SCNG algorithm can be successfully applied to more realistic data. We consider the difficult setting of a piece of Jazz music played by five instruments and recorded by two microphones. Given only the recordings of the microphones the track of each single instrument can be reconstructed even in the presence of some additive gaussian noise.

We compared the performance of SCNG with FastICA. For the comparison, we used the synthetical data as well as the more realistic audio data. In case of the synthetically generated data, we could show that SCNG outperforms FastICA and is more resistent to additive noise. In case of the audio data, SCNG does not outperform FastICA in the case of a low noise level. But if strong additive noise is present, SCNG provides better results.

The results are quite promising and might be improved by using computationally more demanding methods such as basis pursuit for the reconstruction of the sources.

References

[1] Anthony J. Bell and Terrence J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[2] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1998.

[3] David L. Donoho, Michael Elad, and Vladimir N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1):6–18, 2006.

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

[5] A. Hyvärinen. Gaussian moments for noisy independent component analysis. *IEEE Signal Processing Letters*, 6(6):145–147, 1999.

[6] A. Hyvarinen, R. Cristescu, and E. Oja. A fast algorithm for estimating overcomplete ica bases for image windows. *Proceedings of the International Joint Conference on Neural Networks, IJCNN'99*, 2:894–899, 1999.

[7] Aapo Hyvarinen, Juha Karhunen, and Erkki Oja. *Independent Component Analysis*. Wiley-Interscience, May 2001.

[8] Kai Labusch, Erhardt Barth, and Thomas Martinetz. Learning Data Representations with Sparse Coding Neural Gas. In Michel Verleysen, editor, *Proceedings of the 16th European Symposium on Artificial Neural Networks*, pages 233–238. D-Side Publishers, 2008.

[9] Kai Labusch, Erhardt Barth, and Thomas Martinetz. Sparse Coding Neural Gas: Learning of Overcomplete Data Representations. *Neurocomputing*, 72(7-9):1547–1555, 2009.

[10] Te-Won Lee, M.S. Lewicki, M. Girolami, and T.J. Sejnowski. Blind source separation of more sources than mixtures using overcomplete representations. *IEEE Signal Processing Letters*, 6(4):87–90, 1999.

[11] Michael S. Lewicki and Terrence J. Sejnowski. Learning Overcomplete Representations. *Neural Computation*, 12(2):337–365, 2000.

[12] T. Martinetz, S. Berkovich, and K. Schulten. "Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction. *IEEE-Transactions on Neural Networks*, 4(4):558–569, 1993.

[13] T. Martinetz and K. Schulten. A "Neural-Gas Network" Learns Topologies. *Artificial Neural Networks*, I:397–402, 1991.

[14] E. Oja. A simplified neuron model as a principal component analyzer. *J. Math. Biol.*, 15:267–273, 1982.

[15] Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, (381):607–609, 1996.

[16] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, 37(23):3311–3325, 1997.

[17] Y. Pati, R. Rezaiifar, and P. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. *Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems,*, November 1993.

[18] L. Rebollo-Neira and D. Lowe. Optimized orthogonal matching pursuit approach. *IEEE Signal Processing Letters*, 9(4):137–140, 2002.

[19] F. Theis, E. Lang, and C. Puntonet. A geometric algorithm for overcomplete linear ica. *Neurocomputing*, 56:381–398, 2004.