Approaching the Time Dependent Cocktail Party Problem with Online Sparse Coding Neural Gas

Kai Labusch and Erhardt Barth and Thomas Martinetz

University of Lübeck - Institute for Neuro- and Bioinformatics
Ratzeburger Alle 160 23538 Lübeck - Germany

Abstract. We show how the "Online Sparse Coding Neural Gas" algorithm can be applied to a more realistic model of the "Cocktail Party Problem". We consider a setting where more sources than observations are given and additive noise is present. Furthermore, we make the model even more realistic, by allowing the mixing matrix to change slowly over time. We also process the data in an online pattern-by-pattern way where each observation is presented only once to the learning algorithm. The sources are estimated immediately from the observations. In order to evaluate the influence of the change rate of the time dependent mixing matrix and the signal-to-noise ratio on the reconstruction performance with respect to the underlying sources and the true mixing matrix, we use artificial data with known ground truth.

1 Introduction

The problem of following a party conversation by separating several voices from noise focusing on a single voice has been termed the "Cocktail Party Problem". A review is provided in [1]. This problem has been tackled by a number of researchers in the following mathematical setting:

We are given a sequence of observations $\mathbf{x}(1), \ldots, \mathbf{x}(t), \ldots$ with $\mathbf{x}(t) \in \mathbb{R}^N$ that are a linear mixture of a number of unknown sources $\mathbf{a}(1), \ldots, \mathbf{a}(t), \ldots$ with $\mathbf{a}(t) \in \mathbb{R}^M$:

$$\mathbf{x}(t) = C\mathbf{a}(t) \tag{1}$$

Here $C = (\mathbf{c}_1, \ldots, \mathbf{c}_M), \mathbf{c}_j \in \mathbb{R}^N$ denotes the mixing matrix. We may consider the observations $\mathbf{x}(t)$ to be what we hear and the sources $\mathbf{a}(t)$ to be the voices of M persons at time t. The sequence $\mathbf{s}_j = a(1)_j, \ldots, a(t)_j, \ldots$ consists of all statements of person j. Is it possible to estimate the sources $\mathbf{s}_j$ only from the mixtures $\mathbf{x}(t)$ without knowing the mixing matrix C? In the past, a number of methods have been proposed that can be used to estimate the statements $\mathbf{s}_j$ and C when only the mixtures $\mathbf{x}(t)$ are known and one can assume that $M = N$ [2], [3]. Moreover, some methods assume statistical independence of the sources [3].

Unfortunately the number of sources is not always equal to the number of observations, i.e., often $M > N$ holds. Humans have two ears but a large number of persons may be present at a party. The problem of having an overcomplete

set of sources has been studied more recently [4–7]. Due to the presence of a certain amount of additional background noise the problem may become even more difficult. A model that introduces a certain amount of additive noise,

$$\mathbf{x}(t) = C\mathbf{a}(t) + \boldsymbol{\epsilon}(t) \quad \|\boldsymbol{\epsilon}(t)\| \leq \delta, \tag{2}$$

has also been studied in the past [8]. The "Sparse Coding Neural Gas" (SCNG) algorithm [9, 10] can be used to perform overcomplete blind source separation under the presence of noise as shown in [11]. Here we want to consider an even more realistic setting by allowing the mixing matrix to be time dependent:

$$\mathbf{x}(t) = C(t)\mathbf{a}(t) + \boldsymbol{\epsilon}(t) \quad \|\boldsymbol{\epsilon}(t)\| \leq \delta. \tag{3}$$

For the time dependent mixing matrix $C(t) = (\mathbf{c}_1(t), \ldots, \mathbf{c}_M(t)), \mathbf{c}_j(t) \in \mathbb{R}^N$, we require $\|\mathbf{c}_j(t)\| = 1$ without loss of generality. For instance, in the case of the cocktail party setting, this corresponds to party guests who change their position during the conversation. We want to process the observations in an online pattern-by-pattern mode, i.e., each observation is presented only once to the learning algorithm and the sources are estimated immediately. We do not make assumptions regarding the type of noise but our method requires that the underlying sources $\mathbf{s}_j$ are sufficiently sparse, in particular, it requires that the $\mathbf{a}(t)$ are sparse, i.e., only a few persons talk at the same time. The noise level δ and the number of sources M have to be known.

1.1 Source separation and orthogonal matching pursuit

We here briefly discuss an important property of the orthogonal matching pursuit algorithm (OMP) [12] with respect to the obtained performance on the representation level that has been shown recently [13]. It provides the theoretical foundation that allows us to apply OMP to the problem of source separation.

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

Let us assume that we know the mixing matrix $C(t)$ at time t. Let us further assume that we know the noise level δ. Let $\mathbf{a}(t)$ be the vector containing a small number k of non-zero entries such that equation (3) holds for a given observation $\mathbf{x}(t)$. OMP provides an estimation $\mathbf{a}(t)^{\mathrm{OMP}}$ of $\mathbf{a}(t)$ by iteratively constructing $\mathbf{x}(t)$ out of the columns of $C(t)$. Let $C(t)\mathbf{a}(t)^{\mathrm{OMP}}$ denote the current approximation of $\mathbf{x}(t)$ in OMP and $\boldsymbol{\epsilon}(t)$ the residual that still has to be constructed. Let U denote the set of indices of those columns of $C(t)$ 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 $C(t)$ that are indexed by U are denoted by $C(t)^U$. Initially, $\mathbf{a}(t)^{\mathrm{OMP}} = 0$, $\boldsymbol{\epsilon}(t) = \mathbf{x}(t)$ and $U = \emptyset$. OMP works as follows:

1. Select $\mathbf{c}_{l_{\text{win}}}(t)$ by $\mathbf{c}_{l_{\text{win}}}(t) = \arg\max_{\mathbf{c}_l(t), l \notin U}(\mathbf{c}_l(t)^T \boldsymbol{\epsilon}(t))$
2. Set $U = U \cup l_{\text{win}}$
3. Solve the optimization problem $\mathbf{a}(t)^{\text{OMP}} = \arg\min_{\mathbf{a}} \|\mathbf{x}(t) - C(t)^U \mathbf{a}\|_2^2$
4. Obtain current residual $\boldsymbol{\epsilon}(t) = \mathbf{x}(t) - C(t)\mathbf{a}(t)^{\text{OMP}}$
5. Continue with step 1 until $\|\boldsymbol{\epsilon}(t)\| \leq \delta$

It can be shown that

$$\|\mathbf{a}(t)^{\text{OMP}} - \mathbf{a}(t)\| \leq \Lambda_{\text{OMP}}\ \delta \tag{4}$$

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

$$H(C(t)) = \max_{1 \leq i,j \leq M, i \neq j} |\mathbf{c}_i(t)^T \mathbf{c}_j(t)| \tag{5}$$

be the mutual coherence of the mixing matrix $C(t)$. The smaller $H(C(t))$, N/M and k are, the smaller Λ_{OMP} becomes and the smaller $\min(\mathbf{a}(t))$ is allowed to be [13]. Since (4) only holds if the smallest entry in $\mathbf{a}(t)$ is sufficiently large, OMP has the property of local stability with respect to (4) [13]. Furthermore it can be shown that under the same conditions $\mathbf{a}(t)^{\text{OMP}}$ contains only non-zeros that also appear in $\mathbf{a}(t)$ [13]. An even globally stable approximation of $\mathbf{a}(t)$ can be obtained by methods such as basis pursuit [13, 14].

1.2 Optimized Orthogonal Matching Pursuit (OOMP)

The "Sparse Coding Neural Gas" algorithm is based on "Optimised Orthogonal Matching Pursuit" (OOMP) which is an improved variant of OMP[15]. In general, the columns of $C(t)$ are not pairwise orthogonal. Hence, the criterion of OMP that selects the column $\mathbf{c}_{l_{\text{win}}}(t), l_{\text{win}} \notin U$ of $C(t)$ that is added to U is not optimal with respect to the minimization of the residual that is obtained after the column $\mathbf{c}_{l_{\text{win}}}(t)$ has been added. Hence OOMP runs through all columns of $C(t)$ that have not been used so far and selects the one that yields the smallest residual:

1. Select $\mathbf{c}_{l_{\text{win}}}(t)$ such that $\mathbf{c}_{l_{\text{win}}}(t) = \arg\min_{\mathbf{c}_l(t), l \notin U} \min_{\mathbf{a}} \|\mathbf{x}(t) - C(t)^{U \cup l}\mathbf{a}\|$
2. Set $U = U \cup l_{\text{win}}$
3. Solve the optimization problem $\mathbf{a}(t)^{\text{OOMP}} = \arg\min_{\mathbf{a}} \|\mathbf{x}(t) - C(t)^U \mathbf{a}\|_2^2$
4. Obtain current residual $\boldsymbol{\epsilon}(t) = \mathbf{x}(t) - C(t)\mathbf{a}(t)^{\text{OOMP}}$
5. Continue with step 1 until $\|\boldsymbol{\epsilon}(t)\| \leq \delta$

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 $C(t)^U$. R is obtained by removing the projection of the columns of $C(t)$ onto the subspace spanned by $C(t)^U$ from $C(t)$ and setting the norm of the residuals $\mathbf{r}_l$ to one. The residual $\boldsymbol{\epsilon}(t)^U$ is obtained by removing the projection of $\mathbf{x}(t)$ to the subspace spanned by $C(t)^U$ from $\mathbf{x}(t)$.

Initially, $R = (\mathbf{r}_1, \ldots, \mathbf{r}_l, \ldots, \mathbf{r}_M) = C(t)$ and $\boldsymbol{\epsilon}(t)^U = \mathbf{x}(t)$. 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 $\boldsymbol{\epsilon}(t)^U$:

$$l_{\text{win}} = \arg\max_{l, l \notin U} (\mathbf{r}_l^T \boldsymbol{\epsilon}(t)^U)^2 \; . \tag{6}$$

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

$$\mathbf{r}_l = \mathbf{r}_l - (\mathbf{r}_{l_{\text{win}}}^T \mathbf{r}_l) \mathbf{r}_{l_{\text{win}}}, \tag{7}$$

$$\boldsymbol{\epsilon}(t)^U = \boldsymbol{\epsilon}(t)^U - (\mathbf{r}_{l_{\text{win}}}^T \boldsymbol{\epsilon}(t)^U) \mathbf{r}_{l_{\text{win}}} \; . \tag{8}$$

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 $\|\boldsymbol{\epsilon}(t)^U\| \leq \delta$ has been reached, the final entries of $\mathbf{a}(t)^{\text{OOMP}}$ can be obtained by recursively collecting the contribution of each column of $C(t)$ during the construction process, taking into account the normalization of the columns of R in each iteration. The selection criterion (6) ensures that the norm of the residual $\boldsymbol{\epsilon}(t)^U$ obtained by (8) is minimal. Hence, the OOMP algorithm can provide an approximation of $\mathbf{a}(t)$ containing even less non-zeros than the approximation provided by OMP.

2 Learning the mixing matrix

We want to estimate the mixing matrix $C(t) = (\mathbf{c}_1(t), \ldots, \mathbf{c}_M(t))$ from the mixtures $\mathbf{x}(t)$ given the noise level δ and the number of underlying sources M. As a consequence of the sparseness of the underlying sources $\mathbf{a}(t)$, we are looking for a mixing matrix $C(t)$ that minimizes the number of non-zero entries of $\mathbf{a}(t)^{\text{OOMP}}$, i.e., the number of iteration steps required by the OOMP algorithm to approximate $\mathbf{a}(t)$ up to the noise level δ. Furthermore, let us assume that the mixing matrix changes slowly over time such that $C(t)$ is approximately constant for some time interval $[t - T, t]$. Hence, we look for the mixing matrix which minimizes

$$\min_{C(t)} \sum_{t'=t-T}^{t} \|\mathbf{a}(t')^{\text{OOMP}}\|_0 \quad \text{subject to} \quad \|\mathbf{x}(t') - C(t)\mathbf{a}(t')^{\text{OOMP}}\| \leq \delta \; . \tag{9}$$

Here $\|\mathbf{a}(t')^{\text{OOMP}}\|_0$ denotes the number of non-zero entries in $\mathbf{a}(t')^{\text{OOMP}}$. The smaller the norm of the current residual $\boldsymbol{\epsilon}(t')^U$ is, the fewer OOMP iterations have to be performed until the stopping criterion $\|\boldsymbol{\epsilon}(t')^U\| \leq \delta$ has been reached and the smaller $\|\mathbf{a}(t')^{\text{OOMP}}\|_0$ becomes. In order to minimize the norm of the residuals and thereby the expression (9), we have to maximize $(\mathbf{r}_{l_{\text{win}}} \boldsymbol{\epsilon}(t')^U)^2$. Therefore, we consider the following optimization problem

$$\max_{\mathbf{r}_1, \ldots, \mathbf{r}_M} \sum_{t'=t-T}^{t} \max_{l, l \notin U} (\mathbf{r}_l^T \boldsymbol{\epsilon}(t')^U)^2 \quad \text{subject to} \quad \|\mathbf{r}_l\| = 1 \; . \tag{10}$$

We maximize (10) by updating R and $C(t)$ prior to the construction step (7) and (8). The update step in each iteration of the OOMP algorithm is a combination of Oja's learning rule [16] and the Neural Gas [17, 18]. As introduced in [9] one obtains what we called "Sparse Coding Neural Gas" (SCNG) learning rule

$$\Delta \mathbf{r}_{l_k} = \Delta \mathbf{c}_{l_k}(t) = \alpha(t) e^{-k/\lambda(t)} y \left(\boldsymbol{\epsilon}(t)^U - y \, \mathbf{r}_{l_k} \right) \tag{11}$$

with learning rate

$$\alpha(t) = \alpha_0 \left(\alpha_{\text{final}} / \alpha_0 \right)^{t/t_{\max}}, \tag{12}$$

and neighbourhood-size

$$\lambda(t) = \lambda_0 \left(\lambda_{\text{final}} / \lambda_0 \right)^{t/t_{\max}} \tag{13}$$

where

$$-\left(\mathbf{r}_{l_0}^T \boldsymbol{\epsilon}(t)^U\right)^2 \leq \ldots \leq -\left(\mathbf{r}_{l_k}^T \boldsymbol{\epsilon}(t)^U\right)^2 \leq \ldots \leq -\left(\mathbf{r}_{l_{M-|U|}}^T \boldsymbol{\epsilon}(t)^U\right)^2 \;, l_k \notin U \tag{14}$$

and $y = \mathbf{r}_{l_k}^T \boldsymbol{\epsilon}(t)^U$. We have shown [10] that (11) implements a gradient descent with respect to

$$\max_{\mathbf{r}_1,\ldots,\mathbf{r}_M} \sum_{t'=t-T}^{t} \sum_{l=1}^{M} h_{\lambda_t}(k(\mathbf{r}_l, \boldsymbol{\epsilon}(t')^U))(\mathbf{r}_l^T \boldsymbol{\epsilon}(t')^U)^2 \quad \text{subject to} \quad \|\mathbf{r}_l\| = 1, \tag{15}$$

with $h_{\lambda_t}(v) = e^{-v/\lambda_t}$. $k(\mathbf{r}_l, \boldsymbol{\epsilon}(t')^U)$ denotes the number of $\mathbf{r}_j$ with $(\mathbf{r}_l^T \boldsymbol{\epsilon}(t')^U)^2 < (\mathbf{r}_j^T \boldsymbol{\epsilon}(t')^U)^2$, i.e., (15) is equivalent to (10) for $\lambda(t) \to 0$. Due to (11) the updates of all OOMP iterations are accumulated in the learned mixing matrix $C(t)$. Due to the orthogonal projection (7) and (8) performed in each iteration, these updates are pairwise orthogonal. The columns of the original matrix emerge in random order in the learned mixing matrix. The sign of the columns of the mixing matrix $\mathbf{c}_l(t)$ cannot be determined because multiplying $\mathbf{c}_l(t)$ by -1 corresponds to multiplying $\mathbf{r}_l$ by -1 which does not change (15).
What happens for $t > t_{\max}$? Assuming that after $t_{\max}$ learning steps have been performed the current learned mixing matrix is close to the true mixing matrix, we track the slowly changing true mixing matrix by setting $\alpha(t) = \alpha_{\text{final}}$ and $\lambda(t) = \lambda_{\text{final}}$.

3 Experiments

We performed a number of experiments on artificial data in order to study whether the underlying sources can be reconstructed from the mixtures. We consider sequences

$$\mathbf{x}(t) = C(t)\mathbf{a}(t) + \boldsymbol{\epsilon}(t), \; t = 1, \ldots, L, \tag{16}$$

where $\|\boldsymbol{\epsilon}(t)\| \leq \delta$, $\mathbf{x}(t) \in \mathbb{R}^N$, $\mathbf{a}(t) \in \mathbb{R}^M$. The true mixing matrix $C(t)$ slowly changes from state C_{i-1} to state C_i in P time steps. We randomly chose a

sequence of true mixing matrices $C_i,\ i = 1, \ldots, \lceil L/P \rceil$ with entries taken from a uniform distribution. The columns of these mixing matrices were set to unit norm. At time t with $(i-1)P \leq t \leq iP$ the true mixing matrix $C(t)$ is chosen according to

$$C(t) = \left(1 - \frac{(t-(i-1)P)}{P}\right) C_{i-1} + \frac{(t-(i-1)P)}{P} C_i. \tag{17}$$

The norm of the columns of each true mixing matrix $C(t)$ is then set to unit norm. The sources $\mathbf{a}(t)$ were obtained by setting up to k entries of the $\mathbf{a}(t)$ to uniformly distributed random values in $[-1, 1]$. For each $\mathbf{a}(t)$ the number of non-zero entries was obtained from a uniform distribution in $[0, k]$. Uniformly distributed noise $\mathbf{e}(t) \in \mathbb{R}^M$ in $[-1, 1]$ was added such that

$$\mathbf{x}(t) = C(t)(\mathbf{a}(t) + \mathbf{e}(t)) = C(t)\mathbf{a}(t) + \boldsymbol{\epsilon}(t)\ . \tag{18}$$

We want to asses the error that is obtained with respect to the recontruction of the sources. Hence, we evaluate the difference between the sources $\mathbf{a}(t)$ and the estimation $\mathbf{a}(t)^{\mathrm{OOMP}}$ that is obtained from the OOMP algorithm on the basis of the mixing matrix $C^{learn}(t)$ that is provided by the SCNG algorithm:

$$\|\mathbf{a}(t) - \mathbf{a}(t)^{\mathrm{OOMP}}\|_2. \tag{19}$$

With $(\mathbf{s}_1^{\mathrm{OOMP}}, \ldots, \mathbf{s}_M^{\mathrm{OOMP}})^T = (\mathbf{a}(1)^{\mathrm{OOMP}}, \ldots, \mathbf{a}(L)^{\mathrm{OOMP}})$ we denote the estimated underlying sources obtained from the OOMP algorithm. In order to evaluate (19) we have to assign the entries in $\mathbf{a}(t)^{\mathrm{OOMP}}$ to the entries in $\mathbf{a}(t)$ which is equivalent to assigning the true sources $\mathbf{s}_j$ to the estimated sources $\mathbf{s}_j^{\mathrm{OOMP}}$. This problem arises due to the random order in which the columns of the true mixing matrix appear in the learned mixing matrix. Due to the time dependent mixing matrix the assignment may change over time. In order to obtain an assignment at time t, we consider a window of size s_w:

$$(\mathbf{w}_1(t)^{\mathrm{OOMP}}, \ldots, \mathbf{w}_M(t)^{\mathrm{OOMP}})^T = (\mathbf{a}(t - s_w/2)^{\mathrm{OOMP}}, \ldots, \mathbf{a}(t + s_w/2)^{\mathrm{OOMP}}) \tag{20}$$

and

$$(\mathbf{w}_1(t), \ldots, \mathbf{w}_M(t))^T = (\mathbf{a}(t - s_w/2), \ldots, \mathbf{a}(t + s_w/2)). \tag{21}$$

We obtain the assignment by performing the following procedure:

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

$$\frac{|\mathbf{w}_j(t)^{\mathrm{OOMP}} \mathbf{w}_i(t)^T|}{\|\mathbf{w}_i(t)\| \|\mathbf{w}_j(t)^{\mathrm{OOMP}}\|} \text{ is maximal.}$$

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

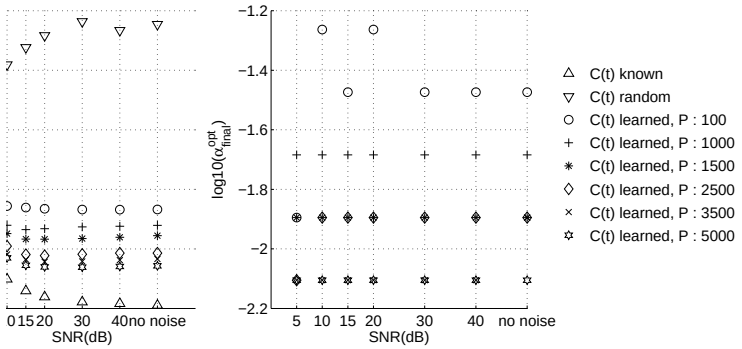

Fig. 1. Left: Mean distance between $\mathbf{a}(t)$ and $\mathbf{a}(t)^{\mathrm{OOMP}}$ for different SNR and P. Right: Best performing final learning rate for each P and SNR.

For all experiments, we used $L = 20000$ and $\alpha_0 = 1$ for the learing rate as well as $\lambda_0 = M/2$ and $\lambda_{\mathrm{final}} = 10^{-10}$ for the neighbourhood size and $t_{\max} = 5000$. We repeated all experiments 20 times and report the mean result over the 20 runs. For the evaluation of the reconstruction error the window size s_w was set to 30 and the reconstruction error was only evaluated in the time interval $t_{\max} < t < L$. In all experiments an overcomplete setting was used consisting of $M = 30$ underlying sources and $N = 15$ available oberservations. Up to $k = 3$ underlying sources were active at the same time.

In the first experiment, we varied the parameter P which controls the change rate of the true mixing matrix as well as the SNR. The final learning rate α_{final} was varied for each combination of P and SNR such that the minimal reconstruction error was obtained. For comparison purposes, we also computed the reconstruction error that is obtained by using the true mixing matrix as well as the error that is obtained by using a random matrix. The results of the first experiment are shown in figure 1. On the left side the mean distance between $\mathbf{a}(t)$ and $\mathbf{a}(t)^{\mathrm{OOMP}}$ is shown for different SNR and P. It can be seen that the larger the change rate of the true mixing matrix (the smaller P) and the stronger the noise, the more the reconstruction performance degrades. But even for strong noise and a fast changing true mixing matrix, the estimation provided by SCNG clearly outperforms a random matrix. Of course, the best reconstruction performance is obtained by using the true mixing matrix. On the right side of the figure the best performing final learning rate for each P and SNR is shown. It can be seen that the optimal final learning rate depends on the change rate of the true mixing matrix but not on the strength of the noise. In order to assess how good the true mixing matrix is learned, we perform an experiment that is similar to an experiment that has been used to asses the performance of the K-SVD algorithm with respect to the learing of the true mixing matrix [19]. Note, that the K-SVD algorithm cannot be applied to the setting that is described in the following. We compare the learned mixing matrix to the true mixinig matrix

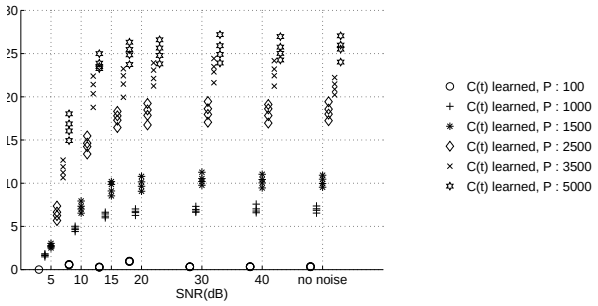

Fig. 2. We sorted the 20 trials according to the number of successfully learned columns of the mixing matrix and order them in groups of five experiments. The figure shows the mean number of successfully detected columns of the mixing matrix for each of the five groups.

using the maximum overlap between each column of the true mixing matrix and each column of the learned mixing matrix, i.e, whenever

$$\max_j \left(1 - |\mathbf{c}_i(t)\mathbf{c}_j^{\text{learn}}(t)|\right) \tag{22}$$

is smaller than 0.05, we count this as a success. We repeat the experiment 20 times with a varying SNR as well as zero noise. For each SNR, we sort the 20 trials according to the number of successfully learned columns of the mixing matrix and order them in groups of five experiments. Figure 2 shows the mean number of successfully detected columns of the mixing matrix for each of the five groups for each SNR and P. The smaller the SNR and the smaller the change rate of the true mixing matrix is, the more columns are learned correctly. If the true mixing matrix changes very fast ($P = 100$) almost no column can be learned with the required accuracy.

4 Conclusion

We showed that the "Sparse Coding Neural Gas" algorithm can be applied to a more realistic model of the "Cocktail Party Problem" that allows for more sources than observations, additive noise and a mixing matrix that is time dependent, which corresponds to persons that change their position during the conversation. The proposed algorithm works online, the estimation of the underlying sources is provided immediately. The method requires that the sources are sparse enough, that the mixing matrix does not change too quickly and that the additive noise is not too strong. In order to apply this algorithm to real-world data, future work is required. The problem of (i) choosing the number of sources M based solely on the observations, (ii) determining the noise level δ based solely on the observations and (iii) obtaining the temporal assignment of the sources based solely on the estimated sources, i.e., thereby not using the sliding window procedure described in the experiments section have to be investigated.

References

1. Haykin, S., Chen, Z.: The cocktail party problem. Neural Comput. **17**(9) (2005) 1875–1902
2. Bell, A.J., Sejnowski, T.J.: An information-maximization approach to blind separation and blind deconvolution. Neural Computation **7**(6) (1995) 1129–1159
3. Hyvärinen, A.: Fast and robust fixed-point algorithms for independent component analysis. IEEE Transactions on Neural Networks **10**(3) (1999) 626–634
4. Hyvärinen, A., Cristescu, R., Oja, E.: A fast algorithm for estimating overcomplete ica bases for image windows. Proceedings of the International Joint Conference on Neural Networks, IJCNN'99 **2** (1999) 894–899
5. Lee, T.W., Lewicki, M., Girolami, M., Sejnowski, T.: Blind source separation of more sources than mixtures using overcomplete representations. IEEE Signal Processing Letters **6**(4) (1999) 87–90
6. Lewicki, M.S., Sejnowski, T.J.: Learning Overcomplete Representations. Neural Computation **12**(2) (2000) 337–365
7. Theis, F., Lang, E., Puntonet, C.: A geometric algorithm for overcomplete linear ica. Neurocomputing **56** (2004) 381–398
8. Hyvärinen, A.: Gaussian moments for noisy independent component analysis. IEEE Signal Processing Letters **6**(6) (1999) 145–147
9. Labusch, K., Barth, E., Martinetz, T.: Learning Data Representations with Sparse Coding Neural Gas. In Verleysen, M., ed.: Proceedings of the 16th European Symposium on Artificial Neural Networks, D-Side Publishers (2008) 233–238
10. Labusch, K., Barth, E., Martinetz, T.: Sparse Coding Neural Gas: Learning of Overcomplete Data Representations. Neurocomputing **72**(7-9) (2009) 1547–1555
11. Labusch, K., Barth, E., Martinetz, T.: Sparse Coding Neural Gas for the Separation of Noisy Overcomplete Sources. In Kurková, V., Neruda, R., Koutník, J., eds.: Artificial Neural Networks - ICANN 2008, 18th International Conference, Prague, Czech Republic, September 3-6, 2008, Proceedings, Part II. Volume 5163 of Lecture Notes in Computer Science., Springer (2008) 788–797
12. Pati, Y., Rezaiifar, R., Krishnaprasad, P.: Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems, (November 1993)
13. Donoho, D.L., Elad, M., Temlyakov, V.N.: Stable recovery of sparse overcomplete representations in the presence of noise. IEEE Transactions on Information Theory **52**(1) (2006) 6–18
14. Chen, S.S., Donoho, D.L., Saunders, M.A.: Atomic decomposition by basis pursuit. SIAM Journal on Scientific Computing **20**(1) (1998) 33–61
15. Rebollo-Neira, L., Lowe, D.: Optimized orthogonal matching pursuit approach. IEEE Signal Processing Letters **9**(4) (2002) 137–140
16. Oja, E.: A simplified neuron model as a principal component analyzer. J. Math. Biol. **15** (1982) 267–273
17. Martinetz, T., Schulten, K.: A "Neural-Gas Network" Learns Topologies. Artificial Neural Networks **I** (1991) 397–402
18. Martinetz, T., Berkovich, S., Schulten, K.: "Neural-gas" Network for Vector Quantization and its Application to Time-Series Prediction. IEEE-Transactions on Neural Networks **4**(4) (1993) 558–569
19. Aharon, M., Elad, M., Bruckstein, A.: K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on] **54**(11) (2006) 4311–4322