MONAURAL SEPARATION AND CLASSIFICATION OF MIXED SIGNALS: A SUPPORT-VECTOR REGRESSION PERSPECTIVE

Sepp Hochreiter and Michael C. Mozer

Department of Computer Science
University of Colorado
Boulder, CO 80309
{hochreit,mozer}@cs.colorado.edu

ABSTRACT

We address the problem of extracting multiple independent sources from a single mixture signal. Standard independent-component analysis approaches fail when the number of sources is greater than the number of mixtures. For this case, the *sparse-decomposition method* [1] has been proposed. The method relies on a dictionary of atomic signals and recovers the degree to which various dictionary atoms are present in the mixture. We show that the sparse-decomposition method is equivalent to a form of support-vector regression (SVR). The training inputs for the SVR are the dictionary atoms, and the corresponding targets are the dot product of the mixture and atom vectors. The SVR perspective provides a new interpretation of the sparse-decomposition method's hyperparameter, and allows us to generalize and improve the method. The most important insight is that the sources do not have to be identical to dictionary atoms, but rather we can accommodate a many-to-one mapping of source signals to dictionary atoms—a classification of sorts—characterized by a known nonlinear transformation with unknown parameters. The limitation of the SVR perspective is that it cannot recover the signal strength of an atom in the mixture; rather, it can only recover whether or not a particular atom was present. In experiments, we show that our model can handle difficult problems involving classification of sources. Our model may be particularly useful for speech signal processing and CDMA-based mobile communication, where in both cases we have knowledge about the invariances in the signal.

1. INTRODUCTION

Independent component analysis (ICA) [2, 3, 4] attempts to recover multiple *source* signals that have been combined into one or more *mixture* signals. Most ICA algorithms assume that the sources are mutually statistically independent and that mixtures are linear combinations of the sources. Well known ICA methods like "infomax" [5], maximum likelihood approaches [6], entropy and cumulant based methods [7, 8, 9] have the restriction that the number of source and mixture signals must be equal. In many real world applications, only one mixture is available. For example, with direct sequence code division multiple access (DS-CDMA) mobile communication, signals from multiple users must be extracted from a single mixture time series. Further, many real world sound recordings (e.g., bird songs, music, traffic, or listening devices used in espionage) use only one or two microphones. Standard ICA approaches cannot be used in these cases.

In contrast, the human auditory system is able to distinguish multiple sound sources from two mixtures—the ears. It can even extract sources from monaural recordings. In some cases, separation of signals is easy because the signals occur in different frequency bands (e.g., bird songs and an oncoming bus), but many times simple physical distinctions are inadequate to recover the signals. Consider a performance by a symphony orchestra. The conductor is able to isolate individual melody lines, instruments, or even musicians from the ensemble, whereas a naive audience member will not. The difference between the conductor and the audience member is the conductor's knowledge and familiarity with the sound patterns that constitute the performance. One could even imagine that the conductor has a dictionary of sound *atoms*—canonical or prototypical musical phrases and timbres—and identification of components comes by isolating the atoms from the mixture.

Several ICA approaches have adopted the idea of using a dictionary to extract multiple sources from fewer or even one mixture [1, 10, 11]. The dictionary can be based on primitive functions (e.g., Fourier bases, wavelet packages, or Gabor functions) [1], predefined based on prior knowledge, or can be trained to fit the problem [12, 11]. Zibulevsky and Pearlmutter [1] specify not only a dictionary, but also a prior that enforces sparseness—i.e., an expectation as to how many sources will be present simultaneously. All these approaches are restricted to mixtures consisting of linearly superimposed dictionary atoms; this restriction is necessary to avoid ambiguity in the problem.

In this paper we show that the *sparse-decomposition method* of Zibulevsky and Pearlmutter can be reinterpreted as ϵ-support vector regression (ϵ-SVR) [13], when there is a single mixture and a Laplacian prior. The ϵ-SVR analogy provides a novel interpretation of the sparse-decomposition method's hyperparameter that determines the degree of sparseness. The analogy also allows one to view the sparse-decomposition method as one member of a family of similar algorithms. Most notably, the family includes a variant of the sparse-decomposition method that allows for nonlinear transformations of the sources before they are mixed, and another nonlinear transformation in the process of identifying dictionary atoms in the mixture.

Applications of the approach include speech recognition, where the dictionary consists of pretrained or typical speech waveforms [12], and DS-CDMA mobile communication, where the dictionary consists of spreading sequences of the users. One benefit of the nonlinearities incorporated into the approach is that—as we will explain in detail later—they can be used to achieve some degree of invariance to irrelevant characteristics of the speech signal. We demonstrate our approach with experiments using noisy single mixtures and speech tasks.

2. SPARSE-DECOMPOSITION VIEWED AS ϵ-SVR

In this section, we review the sparse-decomposition method introduced in [1], focusing on the case of a single mixture signal. We will also describe the relation between sparse decomposition and ϵ-support vector regression (ϵ-SVR).

2.1. The Sparse-Decomposition Method

Denote the mixture signal by $\mathbf{x} \in \mathcal{R}^L$, which—in the case of speech—might correspond to a time series of L discrete time steps. We assume a dictionary matrix, $\mathbf{S} \in \mathcal{R}^{L \times P}$, whose columns consist of the P atomic signals of length L. We assume a generative process in which the mixture is created by first choosing a set of dictionary atoms and then combining them linearly with noise:

$$\mathbf{x} \;=\; \mathbf{S}\,\mathbf{c} + \nu = \sum_{i=1}^{P} c_i \mathbf{s}_i \;+\; \nu \;, \qquad (1)$$

where $\mathbf{c} \in \mathcal{R}^P$ is a vector of weighting coefficients, one per atom, $\nu \sim N\left(0, \sigma^2\right)$ is an L-dimensional i.i.d. additive Gaussian noise vector, and $\mathbf{s}_i$ the ith atom in the dictionary. See Figure 1 for an illustration of the generative process that produces the mixture.

The goal of the sparse-decomposition method is to determine the coefficient vector, $\hat{\mathbf{c}}$, that satisfies two properties: (1) $\mathbf{x}$ must be well approximated by $\hat{\mathbf{c}}$, and (2) $\hat{\mathbf{c}}$ is sparse, i.e., it has elements of small magnitude. These two properties are achieved by a Bayesian approach in which (1) is the likelihood $p\left(\mathbf{x} \mid \mathbf{c}, \mathbf{S}\right)$ and (2) is the prior $p\left(\mathbf{c}\right)$. Thus, the approach tries to maximize the posterior

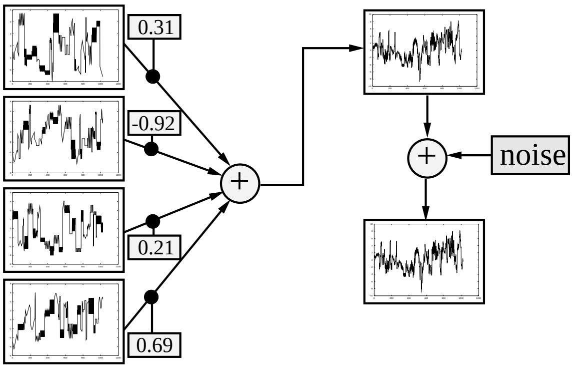

Fig. 1. The data generation process for the sparse-decomposition method. Atoms from the dictionary $\mathbf{S}$ are weighted by a nonnegative factor and added together with noise resulting in the mixture. The goal is to recover the weights from the mixture.

$$p\left(\mathbf{c} \mid \mathbf{x}, \mathbf{S}\right) \;\;\propto\;\; p\left(\mathbf{x} \mid \mathbf{c}, \mathbf{S}\right)\, p\left(\mathbf{c}\right) \;,$$

where we use "$\propto$" because we omit the constant normalization factor in the denominator of Bayes rule. Given the Gaussian noise model, the likelihood is

$$p\left(\mathbf{x} \mid \mathbf{c}, \mathbf{S}\right) \;\;\propto\;\; \exp\left(-\frac{1}{2\,\sigma^2}\left(\mathbf{x} \;-\; \mathbf{S}\,\mathbf{c}\right)^2\right) \;.$$

To enforce sparseness of the coefficients, a Laplacian prior for $\mathbf{c}$ is used with $\|\mathbf{c}\|_1 = \sum_{i=1}^{P} |c_i|$ we have:

$$p\left(\mathbf{c}\right) \;\;\propto\;\; \exp\left(-\frac{\epsilon}{\sigma^2}\,\|\mathbf{c}\|_1\right) \;.$$

Consequently, the posterior is

$$p\left(\mathbf{c} \mid \mathbf{x}, \mathbf{S}\right) \propto \exp\left(-\frac{1}{\sigma^2}\left(\frac{1}{2}\left(\mathbf{x} \;-\; \mathbf{S}\,\mathbf{c}\right)^2 \;+\; \epsilon\;\|\mathbf{c}\|_1\right)\right) .$$

The maximum a posterior (MAP) solution, $\hat{\mathbf{c}}$, is obtained by taking the log of the posterior, flipping its sign, and ignoring irrelevant constant terms and factors, resulting in the minimization problem

$$\hat{\mathbf{c}} = \operatorname{argmin}_{\mathbf{c}} \quad \tfrac{1}{2}\left(\mathbf{x} \;-\; \mathbf{S}\,\mathbf{c}\right)^2 \;+\; \epsilon\; \textstyle\sum_{i=1}^{P} |c_i| \;.$$

By standard techniques, we can turn this unconstrained optimization problem into a constrained optimization problem in which $\mathbf{c}$ is split into two vectors, $\mathbf{c}^+$ and $\mathbf{c}^-$. The MAP solution $\{\hat{\mathbf{c}}^+, \hat{\mathbf{c}}^-\}$ is

$$\begin{aligned} \operatorname{argmin}_{\mathbf{c}^+,\mathbf{c}^-} \quad & \frac{1}{2}\left(\mathbf{c}^+ - \mathbf{c}^-\right)^T \mathbf{S}^T\mathbf{S}\left(\mathbf{c}^+ - \mathbf{c}^-\right) \;- \\ & \mathbf{x}^T\mathbf{S}\left(\mathbf{c}^+ - \mathbf{c}^-\right) \;+\; \epsilon\;\mathbf{1}^T\left(\mathbf{c}^+ + \mathbf{c}^-\right) \\ \text{s.t.} \quad & 0 \leq c_i^+, c_i^- \leq C \;, \qquad (2) \end{aligned}$$

where T is the transposition operator, $\mathbf{1}$ is the vector of ones, and C is an upper bound that can serve as an additional constraint on the solution (which was not part of the original formulation by Zibulevsky and Pearlmutter). In the solution, $\hat{\mathbf{c}} = \hat{\mathbf{c}}^{+} - \hat{\mathbf{c}}^{-}$, allowing us to split the positive and negative elements of $\mathbf{c}$ into two vectors.

We will show that *this formulation has the same core mathematics as ϵ-support vector regression (ϵ-SVR)* [13]. We turn to a brief overview of ϵ-SVR.

2.2. ϵ-Support Vector Regression

ϵ-SVR is a supervised approach to regression in which we are given training data $\{(\mathbf{z}_i, y_i), \ldots, (\mathbf{z}_P, y_P)\}$, where $\mathbf{z}_i \in \mathcal{R}^L$ and y_i is a scalar. The goal is to produce a function, h, such that $h(\mathbf{z}_i)$ closely approximates y_i. In the linear formulation of ϵ-SVR, $h(\mathbf{z}_i) = \langle \mathbf{w}, \mathbf{z}_i \rangle + o$, where $\mathbf{w} \in \mathcal{R}^L$, $o \in \mathcal{R}$, and $\langle ., . \rangle$ denotes the dot product. ϵ-SVR attempts to obtain a "flat" function by minimizing $\frac{1}{2}\|\mathbf{w}\|^2$, but subject to the constraint that the fit is good enough, as quantified by the constraint $|y_i - f(\mathbf{z}_i)| < \epsilon + \xi_i$ for all i. ϵ is a measure of how accurate the fit needs to be, or intuitively, a measure of the noise in the data. The *slack variables* $\xi_i \geq 0$ allow for the fact that it may not be possible to find an h that satisfies the ϵ-accuracy criterion. However, to ensure that the deviations are minimal, the optimization attempts to minimize the magnitude of the slack variables as well; specifically, the constrained optimization is over the objective function $\frac{1}{2}\|\mathbf{w}\|^2 + C\|\xi\|_1$, where C determines the trade off between the flatness of the function and the tolerance of prediction errors.

It turns out that the ϵ-SVR has an alternative but entirely equivalent formulation in which each example i is assigned a coefficient, c_i, and $\mathbf{w}$ is defined with these coefficients: $\mathbf{w} = \sum_{i=1}^{P} c_i\, \mathbf{z}_i$. Consequently, $h(\mathbf{z}) = \sum_{i=1}^{P} c_i \langle \mathbf{z}_i, \mathbf{z} \rangle + o$. The $\mathbf{z}_i$ for which $c_i \neq 0$ are called *support vectors*. In this formulation, learning involves an optimization problem, to search for the $\{\hat{c}_i\}$ that minimize

$$\tfrac{1}{2}\sum_{i,j=1}^{P} c_i\; c_j \langle \mathbf{z}_i, \mathbf{z}_j \rangle \;-\; \sum_{i=1}^{P} y_i\, c_i + \;\epsilon \sum_{i=1}^{P} |c_i|$$

subject to $-C \leq c_i \leq C$. To eliminate the absolute-value function from the objective function, a standard technique is used to split the c_i into positive and negative components, c_i^{+} and c_i^{-}, where $c_i = c_i^{+} - c_i^{-}$, resulting in the optimization problem:

$$\begin{array}{lll}
\min\limits_{\mathbf{c}^{+},\mathbf{c}^{-}} & \dfrac{1}{2}\left(\mathbf{c}^{+} - \mathbf{c}^{-}\right)^{T} \mathbf{Z}^{T}\mathbf{Z}\left(\mathbf{c}^{+} - \mathbf{c}^{-}\right) \; - & \\
 & \mathbf{y}^{T}\left(\mathbf{c}^{+} - \mathbf{c}^{-}\right) \; + \; \epsilon\, \mathbf{1}^{T}\left(\mathbf{c}^{+} + \mathbf{c}^{-}\right) & \\
\text{s.t.} & 0 \leq c_i^{+}, c_i^{-} \leq C & (3) \\
 & \left\|\mathbf{c}^{+} - \mathbf{c}^{-}\right\|_1 = 0 \;, &
\end{array}$$

where the matrix $\mathbf{Z}$ is formed with the $\mathbf{z}_i$ as column vectors. Rather than viewing the constant o as a free parameter of h, the degree of freedom provided by o is used to ensure the second constraint, $\left\|\mathbf{c}^{+} - \mathbf{c}^{-}\right\|_1 = 0$, is satisfied. If the o is set to zero, the constraint drops out of the above optimization formulation.

2.3. The Relationship Between the Sparse-Decomposition Method and ϵ-SVR

Consider data for an ϵ-SVR consisting of P training examples. For example i, the input $\mathbf{z}_i \in \mathcal{R}^L$ is dictionary atom $\mathbf{s}_i$, and the target for the example, y_i, is the dot product between the mixture $\mathbf{x}$ and dictionary atom $\mathbf{s}_i$: $y_i = \langle \mathbf{x}, \mathbf{s}_i \rangle$. If we fix $o = 0$ in this situation, optimization problem (2) is identical to optimization problem (3) because the constraint $\left\|\mathbf{c}^{+} - \mathbf{c}^{-}\right\|_1 = 0$ is eliminated in in (3).

The ϵ-SVR formulation gives an interpretation to the hyperparameter ϵ in the sparse-decomposition method. It is a measure of the noise level in the data, and indirectly affects the number of $\hat{c}_i$ that are significantly non-zero. As depicted in Figure 2, each example will have a target, y_i, that either lies inside or outside the ϵ-*tube*. The closer a target y_i is to zero, the more nearly orthogonal is the mixture $\mathbf{x}$ to atom $\mathbf{s}_i$, and the less likely atom i is to be present in the mixture. Thus, the ϵ-tube distinguishes atoms that are likely to be relevant from those likely to be irrelevant. It turns out that any example i lying outside the ϵ-*tube* will have either $\hat{c}_i = C$ or $\hat{c}_i = -C$. In the sparse-decomposition formulation, $\hat{c}_i$ indicates the degree to which a dictionary atom i is present.

In ϵ-SVR, the parameter o can be nonzero, which extends the sparse-decomposition formalism. To include o in the sparse-decomposition formalism implies that the correlation coefficients y_i have a nonzero mean. o is an approximation for the mean and is estimated by the ϵ-SVR as the expectation over i of $\mathbf{s}_i^T\mathbf{x} - \mathbf{s}_i^T\mathbf{S}\hat{\mathbf{c}} - \epsilon\text{sign}(\hat{c})$. Although we do not explore this possibility further in the present paper, we conjecture that inclusion of o in the sparse-decomposition method may be useful if the dictionary atoms share some structure (e.g., if an atom corresponds to a time series which ends the same way for many atoms) and the mixture also contains this structure, resulting in y_i that are consistently greater than zero.

3. NONLINEAR FORMULATION

In the ϵ-SVR framework, a nonlinear approximation for $\mathbf{y}$ is possible by introducing a kernel function, $k(\mathbf{z}_i, \mathbf{z}_j)$, in the expression for h instead of the dot product $\langle \mathbf{z}_i, \mathbf{z}_j \rangle$. In matrix notation, $\mathbf{Z}^T\mathbf{Z}$ is replaced by the kernel matrix $\mathbf{K}$ with $K_{ij} = k(\mathbf{z}_i, \mathbf{z}_j)$. The purpose of this kernel is to allow each vector z to be mapped into a new representational space, call it the *feature space*, via a function ϕ, where the kernel is defined to be the dot product in the feature space: $k(\mathbf{a}, \mathbf{b}) = \langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle$.

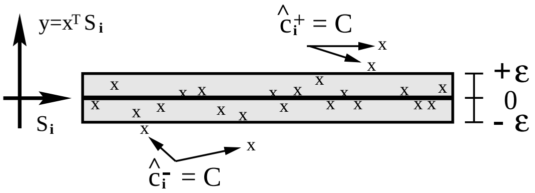

Fig. 2. The linear ϵ–support vector regression corresponding to the sparse-decomposition method. Each "x" in the figure corresponds to a single training example in the ϵ-SVR model. The horizontal axis is a one-dimensional depiction of the input space, and the vertical axis is the target output. The grey area, the ϵ-tube, specifies the range of target outputs that are not significantly different from zero. The examples i that lie outside the ϵ-tube will have $|\hat{c}_i| = C$.

In the following sections, we introduce two nonlinear generalizations of the sparse-decomposition method. In both cases, the nonlinearity involves the mapping of the dictionary atoms $\mathbf{s}_i$ into a feature space. One can conceive of this mapping as part of the generative process that turns a class template into class instances (e.g., a canonical word into a word spoken by a particular speaker at a particular rate).

3.1. Nonlinear Approximation of the Linear Correlations

In the first nonlinear approach, we replace $\mathbf{S}^T\mathbf{S}$ by kernel matrix $\mathbf{K}$. Because of Equation (1), we can recover the MAP coefficients $\hat{\mathbf{c}}$ from coefficients, denoted $\hat{\kappa}$, obtained from the MAP solution to the nonlinear problem:

$$\hat{\mathbf{c}} \;\approx\; \left(\mathbf{S}^T\mathbf{S}\right)^{-1}\mathbf{K}\,\hat{\kappa} \;-\; \left(\mathbf{S}^T\mathbf{S}\right)^{-1}\mathbf{S}^T\mathbf{d}\,,$$

where the "$\approx$"-sign indicates an approximation because ϵ and the noise vector ν are present. $\hat{\kappa_i} = C$ or $\hat{\kappa_i} = -C$ indicates the presence of dictionary atom i in the mixture (see Figure 3).

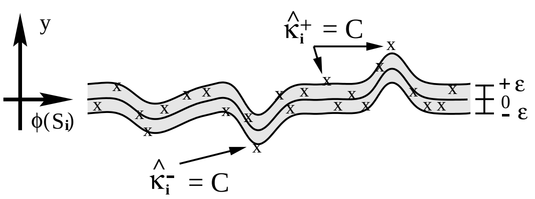

Fig. 3. The nonlinear ϵ-SVR. The dictionary atoms $\mathbf{s}_i$ are mapped by ϕ into a feature space.

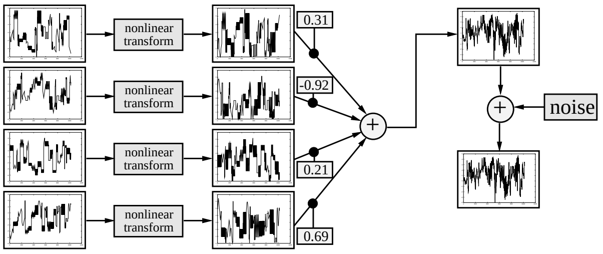

Fig. 4. The data generation process with atom transformations. In contrast to the sparse-decomposition method (see Figure 1) the atomic sources are transformed before they get superimposed.

3.2. Transformed Atomic Sources

Each dictionary atom $\mathbf{s}_i$ is transformed by a mapping f_i (from $\mathcal{R}^L$ into $\mathcal{R}^L$) before it gets superimposed with other transformed atoms to generate the mixture (see Figure 4). Equation (1) becomes

$$\mathbf{x} \;=\; \sum_{j=1}^{P} \kappa_i\; f_i\left(\mathbf{s}_i\right) \;+\; \nu\;.$$

Let $f(\mathbf{S})$ be the transformed dictionary, that is the matrix with vector components $\left[f(\mathbf{S})\right]_i = f_i\left(\mathbf{s}_i\right)$. We minimize

$$\left| f_i\left(\mathbf{s}_i\right)^T \mathbf{x} \;-\; f_i\left(\mathbf{s}_i\right)^T f\left(\mathbf{S}\right)\kappa \right|_{\epsilon}\;.$$

We assume f_i is known except for a free parameter k_i, where $f_i(\mathbf{a}) = g(\mathbf{a}; k_i)$. This implies that all f_i stem from one class of functions. Further we assume that functions from this class do not change certain features of the original atoms. The transformation is invariant with respect to a special feature vector, produced by ϕ. That is,

$$\phi\left(f_i\left(\mathbf{s}_i\right)\right) \;=\; \phi\left(\mathbf{s}_i\right)\;.$$

Our goal is to approximate the mixture feature $\phi\left(\mathbf{x}\right)$ linearly by the atom features $\phi\left(\mathbf{s}_i\right)$ in the feature space. Thus, the mixture feature is assumed to be the weighted sum of the atom features for atoms that are present in the mixture. That can be formulated as

$$\phi\left(\mathbf{x}\right) \;=\; \sum_{i=1}^{P} \kappa_i \phi\left(f_i\left(\mathbf{s}_i\right)\right) \;=\; \sum_{i=1}^{P} \kappa_i \phi\left(\mathbf{s}_i\right)\;.$$

The most notable fact is that the unknown transformations f_i are removed from our approximation problem. $\hat{\kappa_i}$ indicates whether mixture $\mathbf{x}$ and atom $\mathbf{s}_i$ share the same features or not: large $\hat{\kappa_i}$ implies that $\mathbf{x}$ and $\mathbf{s}_i$ are mapped to similar (correlated) feature vectors.

4. EXPERIMENTS

4.1. Nonlinear Approximation of the Linear Correlations

We use a dictionary consisting of 64 atoms of length 128, interpreted as a periodic time series. The atoms vary in their frequency and shape (e.g., sinusoidal, triangular, rectangular, and asymmetric triangular). On average we choose 4.5 dictionary atoms per mixture. The nonzero coefficients c_i are randomly chosen from $[0.1, 1.0] \cup [-1.0, -0.1]$. We added Gaussian noise with variance 1.0 to each mixture component, added sinusoidal noise to the mixture with amplitude from $[0, 0.2]$, and introduced phase shifts to each atom (randomly 0-20 % of the period). The values for hyperparameters ϵ and C are chosen through a validation set. They are adjusted so that fewer than 0.5 sources fail to be recognized per problem. By doing so, we place an upper bound on false rejects (failing to detect sources that are present). To evaluate performance, we measure false accepts (the number of sources detected that were not present in the mixture).

The linear sparse-decomposition method leads to an error of 35.32. Table 1 shows the result for the nonlinear kernel $k(\mathbf{a}, \mathbf{b}) = (q + \mathbf{a} \cdot \mathbf{b})^p$ with various values for p and q. The nonlinear kernel leads to better results than the linear sparse-decomposition method. This demonstrates that nonlinear kernels work even for linear problems.

$p \setminus q$	10^4	10^5	10^6	10^7
2	35.14	35.21	35.23	35.23
5	35.22	35.20	35.23	35.23
10	35.30	35.20	35.22	35.23
20	35.25	35.14	35.21	35.23
30	35.40	35.19	35.21	35.23

Table 1. Average (over 100 trials) of wrongly detected atoms for kernels $k(\mathbf{a}, \mathbf{b}) = (q + \mathbf{a} \cdot \mathbf{b})^p$.

4.2. Transformed Atomic Sources

4.2.1. *Artificial Data*

We consider the following class of componentwise transformations of the dictionary atoms: $[f_i(\mathbf{a})]_j = |a_j|^{w_i}$, which produce components ≥ 0. We note that the local variance remains the same even if negative values are transformed into positive values. To compute the local variance around a point j with window l, we use
$(2l + 1)^{-1} \sum_{t=j-l}^{j+l} (a_t - \bar{a}_j)^2$, where
$\bar{a}_j := (2l + 1)^{-1} \sum_{t=j-l}^{j+l} a_t$. We used 3 different values for l: 8, 10, and 20 (which we denote as conditions AV8, AV10, and AV20).

We generated 100 dictionary atoms of length 1024. To produce an atom we segmented the 1024 length vector into random segments of length between 1 and 64. Each segment consists of a scaled (from $[-0.8, -0.2] \cup [0.2, 0.8]$) periodic function from the previous experiment. A constant between $[-\gamma, +\gamma]$ is added to each segment. Figure 4 depicts the data generation process.

As the transformation, task 1 uses $|a_j|$, task 2 uses a_j^2, and task 3 uses $|a_j|^w$ with w randomly selected from $[0.5, 2.0]$. For task 1, $\gamma = 5.0$, and for tasks 2 and 3, $\gamma = 0.5$. The transformations are mixed as in previous experiment and Gaussian noise is added with $\sigma = 0.01$.

As in the previous experiment, we set parameters to ensure a false reject rate below 0.4 sources per example. The false accept rates are shown in Table 2. The linear model—equivalent to the sparse-decomposition method—failed. The nonlinear mapping by the local variance formulas was able to extract the invariant and, therefore, to classify an atom as being present or not.

	linear	AV8	AV10	AV20
task 1: a_i^2	failed	0.63	0.72	0.99
task 2: $\vert a_i \vert$	failed	5.41	7.38	failed
task 3: $\vert a_i \vert^w$	failed	0.55	0.84	2.37

Table 2. Average (over 100 mixtures) number of wrongly detected atoms for the linear sparse-decomposition method ("linear") and three different methods measuring the local variance (AV8, AV10, and AV20).

4.2.2. *Speech Data*

We considered transformations which shift individual dictionary atoms. As an invariant we use the power spectrum. The dictionary entries are 5 spoken words ("hid", "head", "had", "hud", and "hod") spoken by 20 different speakers (dictionary size is 100). The data was obtained from `areas/speech/database/hvd/` in the AI-Repository at `cs.cmu.edu`. The speech is sampled at 10kHz.

We did not restrict the shifts of the atoms. The coefficients c_i are chosen from $[0.2, 0.8]$. The power spectrum is obtained by using fast Fourier transformation with shifting Hamming window of size 256. The 20 lowest frequencies were set to zero. The additive Gaussian noise had standard deviation of $\sigma = 0.05$ for task 1 and $\sigma = 0.2$ for task 2.

We compared three methods: PS1 is the power spectrum of the original mixture, PS2 is the power spectrum of the mixture where absolute mixture values smaller than 0.05 are set to zero, and PS3 is the power spectrum of the mixture where absolute mixture values smaller that 0.1 are set to zero. As in previous experiments, we set parameters to fix the maximum false reject rate, to 0.64 for task 1 and 0.74 for task 2.

The linear approach completely failed to solve the task. The false accept rate for the nonlinear power-spectrum transformation is shown in Table 3, for the two tasks and the three methods. Figure 5 shows an example of PS3.

	linear	PS1	PS2	PS3
Task 1	failed	1.82	1.72	1.50
Task 2	failed	5.06	4.82	5.10

Table 3. Average number of wrongly detected atoms for the linear sparse-decomposition method ("linear"), and three nonlinear transformation into the power spectrum ("PS1" to "PS3"). The error values are an average of 100 mixtures.

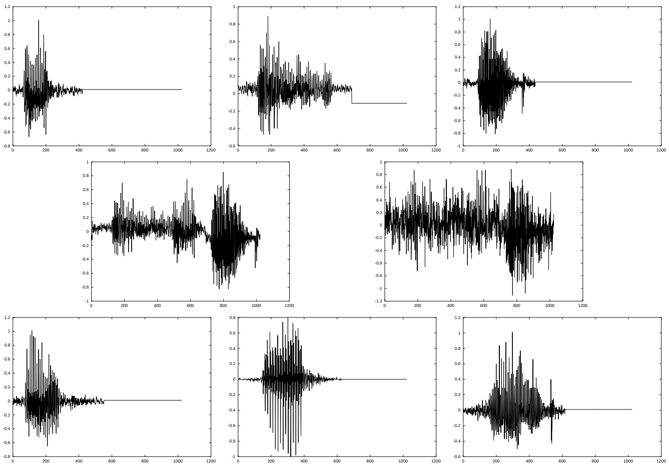

Fig. 5. Example of method PS3, which detected six dictionary atoms in the mixture, three correctly and three incorrectly. Row 1: three dictionary atoms present (but shifted in time) in the mixture. Row 2: the mixture before (left) and after (right) noise is added. Row 3: the three wrongly detected atoms (of 100).

5. CONCLUSION

In this paper we reinterpreted the sparse-decomposition method for a single mixture as ϵ-support vector regression (ϵ-SVR). The ϵ-SVR analogy supplied a new view on the sparse-decomposition method's hyperparameter and allowed to introduce family of similar algorithms of which the sparse-decomposition method is one member. This family includes methods that allow for nonlinear transformations of the sources before they get mixed, and another nonlinear transformation in the process of identifying dictionary atoms in the mixture. One benefit of the nonlinearities incorporated into the approach is that they can be used to achieve some degree of invariance to irrelevant characteristics of signals. We demonstrated our approach with experiments using noisy single mixtures and a speech dataset.

Acknowledgments

This work was supported by the *Deutsche Forschungsgemeinschaft* (Ho 1749/1-1).

6. REFERENCES

[1] M. Zibulevsky and B. A. Pearlmutter, "Blind source separation by sparse decomposition," *Neural Computation*, vol. 13, no. 4, pp. 863–882, 2001.

[2] A. Cichocki, R. Unbehauen, L. Moszczynski, and E. Rummert, "A new on-line adaptive algorithm for blind separation of source signals," in *Proc. Int. Symposium on Artificial Neural Networks, ISANN-94*, 1994, pp. 406–411.

[3] A. Hyvärinen, "Survey on independent component analysis," *Neural Computing Surveys*, vol. 2, pp. 94–128, 1999.

[4] C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, no. 1, pp. 1–10, 1991.

[5] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural Computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[6] H. Attias and C. E. Schreiner, "Blind source separation and deconvolution: The dynamic component analysis algorithm," *Neural Computation*, vol. 10, no. 6, pp. 1373–1424, 1998.

[7] S. Amari, A. Cichocki, and H.H. Yang, "A new learning algorithm for blind signal separation," in *Advances in Neural Information Processing Systems 8*, David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, Eds. 1996, pp. 757–763, MIT Press, Cambridge, MA.

[8] P. Comon, "Independent component analysis – a new concept?," *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[9] J.-F. Cardoso and A. Souloumiac, "Blind beamforming for non Gaussian signals," *IEE Proceedings-F*, vol. 140, no. 6, pp. 362–370, 1993.

[10] G. Cauwenberghs, "Monaural separation of independent acoustical components," in *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems (ISCAS'99)*. 1999, vol. 5, pp. 62–65, IEEE.

[11] T.-W. 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*, 1998.

[12] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Computation*, vol. 12, no. 2, pp. 337–365, 2000.

[13] V. Vapnik, *The nature of statistical learning theory*, Springer-Verlag, New York, 1995.