Learning Efficient Data Representations with Orthogonal Sparse Coding

Henry Schütze, Erhardt Barth, Member, IEEE, and Thomas Martinetz, Senior Member, IEEE

Abstract—We present the learning algorithm Orthogonal Sparse Coding (OSC) to find an orthogonal basis in which a given data set has a maximally sparse representation. OSC is based on stochastic descent by Hebbian-like updates and Gram-Schmidt orthogonalizations, and is motivated by an algorithm that we introduce as the Canonical Approach (CA). First, we evaluate how well OSC can recover a generating basis from synthetic data. We show that, in contrast to competing methods, OSC can recover the generating basis for quite low and, remarkably, unknown sparsity levels. Moreover, on natural image patches and on images of handwritten digits, OSC learns orthogonal bases that attain significantly sparser representations compared to alternative orthogonal transforms. Furthermore, we demonstrate an application of OSC for image compression by showing that the rate-distortion performance can be improved relative to the JPEG standard. Finally, we demonstrate state of the art image denoising performance of OSC dictionaries. Our results demonstrate the potential of OSC for feature extraction, data compression, and image denoising, which is due to two important aspects: (i) the learned bases are adapted to the signal class, and (ii) the sparse approximation problem can be solved efficiently and exactly.

Index Terms—Sparse coding, sparse representation, dictionary learning, blind source separation, sparse component analysis, orthogonal mixture, transform coding, image compression, image denoising.

I. INTRODUCTION

According to the efficient-coding hypothesis, early work on sparse coding proposed that the goal of visual coding is to faithfully represent the visual input with minimal neural activity, an idea that goes back to Barlow [1] and is based on earlier work of Ernst Mach and Donald MacKay. This principle of efficient coding has been later extended in several ways and related to the statistics of natural images [2], [3], [4]. Natural images occupy only a small fraction of the entire signal space. As a consequence, they can indeed be encoded sparsely, meaning that they can be represented by a linear combination of rather few elementary signals out of a given collection. Sparsity can also be observed in other classes of natural signals, for instance acoustic signals [5].

The fact that natural images can be sparsely encoded has already been used for image compression. By choosing an adequate analytic transform, e.g. the Discrete Cosine Transform (DCT) or suitable wavelets, many transform coefficients are small and thus need not be encoded [6], [7]. An important progress has been made by going from such predefined transforms to dictionaries that are learned and thereby adapted to particular signal classes [4]. However, such learned dictionaries have not yet been widely used for image compression, one reason being that the resulting non-orthogonal dictionaries are more complex and computationally more demanding than orthogonal ones.

A. Overcomplete vs. Orthogonal Dictionaries

Learning a dictionary for sparse coding can be seen as the task to identify multiple linear subspaces the given training samples are contained in. A dictionary is a collection of unit length vectors, called atoms, such that any \( K \)-subset of it spans a \( K \)-dimensional subspace of the input space. Suppose a data sample can be represented by some given dictionary via a sparse coefficient vector having \( K \) non-zero entries. These coefficients then correspond to coordinates in a particular subspace that is specified by the support of the coefficient vector and the corresponding atoms.

Learning overcomplete dictionaries allows to arbitrarily increase the collection of atoms to a size larger than the dimensionality of the signal space which in turn increases the number of possible subspaces that can be used for encoding. Subspaces composed from an overcomplete dictionary are mutually non-orthogonal which, in general, enables a better adaptation to the training data set and can “represent a wider range of signal phenomena” [8]. However, not to require further conditions on the dictionary is problematic when it comes to calculating optimal sparse data representations, i.e., optimal coefficient vectors including their support. For general overcomplete dictionaries, this problem is provably NP-hard [9]. Approximative greedy algorithms like Basis Pursuit or Orthogonal Matching Pursuit can find optimal coefficients only if the dictionary obeys particular incoherence properties such as, for instance, the restricted isometry property [10]. These incoherence properties require that dictionary atoms are not too similar and can be seen as a relaxation of orthogonality. However, unlike orthogonality, it is difficult to embed such properties as constraints in dictionary learning algorithms.

Orthogonal dictionaries, on the other hand, are mathematically simple and, additionally, maximally incoherent. All possible subspaces are mutually orthogonal with the implication that optimal coefficients can be calculated simply by inner products. Moreover, an orthogonal dictionary can be easily inverted. It serves as synthesis operator and its transpose as analysis operator. Nevertheless, orthogonal bases learned for sparse coding are able to provide efficient encodings as will be shown by our numerical experiments.
B. Learning an Orthogonal Basis for Sparse Coding

In the following, we address the task of learning an orthogonal basis that provides an optimal $K$-sparse representation of a given training data set. We define an orthogonal basis as a real matrix $U \in \mathbb{R}^{N \times N}$ obeying $U^T U = I_N$. Given such a basis, any signal $x \in \mathbb{R}^N$ has a unique representation $x = Ua$ by the coefficient vector $a \in \mathbb{R}^N$. A signal $x$ is said to be sparse with respect to $U$, if many coefficients, i.e., many entries of $a$, are zero or close to zero. Suppose $U$ is given, then the optimal $K$-sparse coefficient vector of a single sample $x$ is found as a solution of

$$a^*_U(x) = \arg \min_{\|a\|_0 \leq K} \|x - Ua\|_2^2,$$

where $\|\cdot\|_0 : \mathbb{R}^N \rightarrow \{0, \ldots, N\}$ counts the number of non-zero entries. Due to orthogonality and completeness of $U$ an optimal solution $a^*_U(x)$ can be easily determined by keeping the $K$ largest entries $|a_n|$ of $a = U^T x$ and setting the remaining entries to zero. The sparse approximation problem (1) is a subproblem of our task to find the best orthogonal sparse coding basis $U^*_K(X)$. Let $X = (x_1, \ldots, x_L)$ be a given training data set and $A^*_U(x)$ a matrix that column-wise contains a solution to (1) for each $x_i$. Finding $U^*_K(X)$ is a nested optimization problem defined by

$$U^*_K(X) = \arg \min_{U \in O(N)} \|X - UA^*_U(x)\|_F^2,$$

where $O(N) = \{U \in \mathbb{R}^{N \times N} : U^T U = I_N\}$ denotes the orthogonal group.

C. Related Work

1) Analytic Transform Design: The problem of finding a suitable (universal) signal transform in order to efficiently encode image patches and to compress images can be traced back to the Fourier Transform and local versions thereof [11] that finally converged to the first JPEG standard [6] based on the DCT. Pioneering work in the field of wavelet analysis [12] led to a signal decomposition scheme [13], [14] that provides orthogonal multiscale transforms simply by translating and dilating an elementary function (see, e.g., [15], [16]). Further findings in the field of wavelet theory set the stage to more efficient image compression codecs such as JPEG2000 [7].

2) Learning Overcomplete Dictionaries: Olshausen and Field introduced the sparse coding principle and proposed a batch learning algorithm to learn a redundant dictionary that minimizes a regularized joint cost function composed of a representation error term and a term that promotes the sparsity of the representation [17]. Meanwhile, a number of algorithms have been proposed that can learn overcomplete dictionaries. Lewicki and Sejnowski proposed a probabilistic approach by gradient ascent on a log posterior with respect to the dictionary [18]. The authors also deduced that learning an overcomplete sparse coding dictionary is a generalization of Independent Component Analysis (ICA) [19], a method designed to invert linear mixtures of statistically independent source signals by maximizing the marginal non-Gaussianity of the data representation [20]. Aharon et al. proposed K-SVD [21], an algorithm that generalizes K-means clustering and iterates two alternating stages. In the first stage, a pursuit algorithm approximates the optimal K-sparse representations of the training set. In the second stage, each dictionary atom as well as associated coefficients are sequentially updated via Singular Value Decomposition (SVD) of a particular error matrix. Labusch et al. proposed Sparse Coding Neural Gas (SCNG) [22], a soft-competitive online learning algorithm based on Neural Gas [23] clustering. SCNG performs stochastic gradient descent and similarly alternates between updating the sparse coefficients and updating the dictionary atoms. Lesage et al. proposed to learn an overcomplete sparse-coding dictionary as unions of orthogonal bases [24], because it relieves the sparse approximation problem. Alternative approaches to learn overcomplete dictionaries for sparse coding can be found in [25], [26], [27], [28], [29] to name a few.

3) Learning Orthogonal Dictionaries: Nevertheless, a few authors proposed to learn orthogonal dictionaries for sparse coding. Coifman et al. proposed the Wavelet Packet Transform [30], which is an early attempt to enhance orthogonal transforms with a certain degree of adaptivity to the represented signal. For a given signal, it allows to select a basis from a large collection of dyadic time frequency atoms derived from a specific pair of mother wavelet and scaling function. Mishali and Eldar formulated the orthogonal sparse coding problem as a blind source separation problem [31]. Given a set of observations, the problem is to invert the linear mixture of unknown $K$-sparse sources by an unknown orthogonal mixing matrix. They proposed a method with two succeeding stages. The first stage estimates the entire support pattern of the coefficient matrix, i.e., all locations of non-zero coefficients. The second stage iteratively adapts (i) the non-zero coefficients and (ii) the orthogonal mixing matrix via SVD by using the estimated support pattern. However, in [31] only low-dimensional synthetic data sets were investigated, and merely two rather high sparsity levels ($K \in \{2, 3\}$) were considered. With lower sparsity levels the recovery performance of the support pattern rapidly decreases and impairs the estimation of the mixing matrix substantially.

Another issue with their first stage is the strict requirement that the given data need to have an exactly $K$-sparse representation (with exact zeros) in the generating basis, which does not even tolerate small amplitude noise and is therefore not applicable to real world data. For this reason, we can evaluate this approach only with noiseless synthetic data (see Subsection III-A).

Dobigeon and Tourneret proposed the Bayesian framework BOCA for a similar source separation formulation that is, however, designed for an undercomplete orthogonal mixing matrix [32]. BOCA relies on knowing specific prior distributions for the unknown model parameters. The proposed approach models the sparse sources as Bernoulli-Gaussian processes and uses a uniform prior distribution on the Stiefel manifold for the mixing matrix. A comparison of OSC or CA to BOCA is out of the scope of this paper, because we here address the
complete orthogonal dictionary learning task.

Gribonval and Schnass considered the problem of learning an orthogonal sparse-coding basis by minimizing the \( \ell_1 \)-norm of the coefficient matrix with respect to dictionary and coefficient matrix such that their product synthesizes the training data [33]. Their main results are identifiability conditions that guarantee local convergence to the generating dictionary by the \( \ell_1 \) minimization approach. They showed that a particular sparse model satisfies these conditions with high probability provided that enough samples are given. However, an explicit algorithm is not proposed and the convergence relies on a good initialization.

In [34], Bao et al. proposed a batch algorithm to learn an orthogonal sparse coding basis from patches of an image. Their method is similar to the method proposed in [35], but learns the orthogonal dictionary within the orthogonal complement of a predefined (possibly empty) set of orthogonal dictionary atoms which are not updated during learning. Their method is related to what we will introduce as Canonical Approach (CA) in Subsection II-A, as it computes closed form solutions of the two underlying subproblems. However, [34] addresses an unconstrained sparse model in Lagrangian form with a weighting factor \( \lambda \) that controls the trade-off between reconstruction error and sparsity. The dictionary update stage of [34] and CA is the same, and has been used previously, e.g., in [24] and [31]. The key difference between [34] and CA is the sparse coding stage, which is in one case realized by a global hard thresholding operator and in the other case realized by retaining the \( K \) most relevant coefficients for each sample. Since we are focused on the constrained \( K \)-sparse model, we have not included the threshold based method proposed in [34] in our experiments. In [34], the main learning algorithm has also been extended to learn orthogonal dictionaries on corrupted image patches with the objective to better solve image restoration problems. Furthermore, [36] has adopted the method proposed in [34] to solve a compressive MRI imaging reconstruction problem.

To the best of our knowledge there are no further methods which address problem (2) or can serve as a reference for the experiments described in Section III.

D. Structure and Contribution of the Paper

In Section II, we first introduce the batch algorithm CA (which is related to prior work) and subsequently, as our main contribution, the online algorithm OSC to learn an orthogonal sparse coding basis. For OSC, we provide a pseudo code and estimate its computational complexity. We provide a theorem which states that an OSC update reduces the cost for the processed sample. This assures stochastic descent on the cost function for the entire training data set and, hence, cost reduction at least to a local minimum. We demonstrate it in a simulation.

In Section III, we compare the performance of OSC and alternative methods in terms of how well they recover a generating orthogonal basis from sparse synthetic data in noisless and noisy settings. We then apply OSC to real world data, (i) natural image patches, and (ii) images of hand-written digits and visualize the learned orthogonal dictionary atoms. Signal classes different from natural image patches were marginally considered by prior work on orthogonal dictionary learning. Furthermore, the optimal \( K \)-term approximation performance is analyzed on corresponding test data sets for different parameter values \( K \) and compared to alternative orthogonal transforms (analytic as well as learned).

In Section IV, we demonstrate the applicability of OSC to image compression. We compare the rate distortion for images compressed with JPEG, JPEG2000, and a modified JPEG codec for which the \( 8 \times 8 \) DCT has been replaced by an \( 8 \times 8 \) OSC basis. Finally, we use learned orthogonal sparse coding bases for image denoising.

II. THE OSC ALGORITHM

A. The Canonical Approach and Motivation for OSC

First of all, we propose what we call the Canonical Approach (CA) to learn an orthogonal sparse coding basis. CA is related to orthogonal dictionary learning approaches proposed in [34] and [35], CA can be seen as their natural modification to match the problem formulation (2). Similar to traditional sparse coding algorithms, this approach iteratively alternates between (i) the determination of the optimal (column-wise) \( K \)-sparse coefficient matrix \( A_{U,K}(X) \) for the current, temporary fixed basis \( U \), and (ii) the determination of the optimal orthogonal basis \( U_{\lambda}(X) \) for the current, temporary fixed coefficient matrix \( A \). For our orthogonal setting, both subproblems have closed form solutions.

For a temporary fixed \( U \) and a given training sample \( x \), the minimizer of the sparse approximation problem (1) is

\[
a_{U,K}^*(x) = D_K(x, U)U^T x,
\]

where \( D_K(x, U) \) is a diagonal matrix having \( K \) entries equal to 1 and otherwise entries equal to 0. \( D_K(x, U) \) selects the \( K \) largest projections \( |u_i^T x| \), i.e., the \( K \) largest entries \( |a_i| \) of \( a = U^T x \). This step, applied to each training sample \( x_i \), provides an optimal coefficient matrix \( A = A_{U,K}^*(X) \) with \( K \)-sparse columns.

Suppose such a coefficient matrix \( A \) (representing \( X \)) is now temporary fixed, then the optimal orthogonal basis is given by

\[
U_{\lambda}^*(X) = V W^T,
\]

where \( V \) and \( W \) are the outer matrices of the SVD of \( X A^T = V \Sigma W^T \) [37]. Please note that variants of (4) are also used, for instance, in [24], [31], [34], and [35]. Setting \( U = U_{\lambda}^*(X) \) as given by (4) and iterating all these steps yields the CA algorithm.

We tested CA on synthetic \( K \)-sparse data for which the ground truth is known (see Subsection III-A). It turns out that, given the correct sparsity level \( K \), CA is able to accurately recover the generating basis as far as \( K \) is rather small (high sparsity). However, in the case of larger \( K \) (lower sparsity) CA gets stuck in local minima and does not converge. This finding motivates the stochastic descent approach that will be presented in the following.
B. OSC Idea and Pseudo code

Orthogonal Sparse Coding (OSC\(^1\)) can be seen as a generalization of Principal Component Analysis (PCA). With (3) as the solution to the sparse approximation problem (1), the cost function which has to be minimized in (2) can be reformulated as

\[
E_{X,K}(U) = -\sum_{i=1}^{L} x_i^T U D_K(x_i, U) U^T x_i .
\] (5)

We denote the minimizer of (5) by \(U^*_K(X)\). Note that in (5) we have merged both subproblems to a certain extent, a procedure that has been proven to be also crucial for good convergence of online PCA algorithms. If \(D_K\) were a fixed matrix and not dependent on \(x_i\) and \(U\), e.g. the first \(K\) values in the diagonal would be constantly set to one, then \(PCA\) would provide a minimizer of (5). In contrast to \(PCA\), however, \(OSC\) selects for each \(x_i\) an individual \(D_K\) depending on \(U\).

Since the minimizer of (5) cannot be obtained in closed form, we perform stochastic descent analogous to online PCA algorithms. When adapting \(U\), the algorithm switches between gradient descents (Hebbian-like updates) and Gram-Schmidt orthogonalization steps. The OSC algorithm starts with a randomly initialized basis \(U\). For each learning step an entire update of \(U\) takes place. At the beginning of each learning step \(t\), the current learning rate \(\epsilon_t\) is set and a training sample \(x\) is randomly selected from the given training data set \(X\). First, OSC determines an index sequence \(n_1, ..., n_N\) that sorts the squared coefficients of \(x\) in the current basis \(U\), s.t. \((u_{n_1}^T x)^2 \geq (u_{n_2}^T x)^2 \geq ... \geq (u_{n_N}^T x)^2\). Note that the contribution of the selected sample \(x\) to the cost function (5) is given by

\[
E_K(x, U) = -x^T U D_K(x, U) U^T x = -\sum_{k=1}^{K} (u_{n_k}^T x)^2 .
\] (6)

As a consequence, costs are reduced if the sum of the \(K\) largest squared coefficients is increased. Loosely speaking, this can be interpreted as concentrating the energy distribution of the sparse coefficients of \(x\). The determined index sequence defines the order of basis vector updates, starting with the basis vector which contributes most to (6). Before a basis vector \(u_{n_{k}}\) is updated, it is orthogonalized with respect to \(span\{u_{n_1}, ..., u_{n_{k-1}}\}\), the span of basis vectors that were already updated during the current learning step. Then, the orthogonalized basis vector \(u_{n_k}\) is updated by gradient descent depending on the residual vector \(x_{res}\) (the original training sample \(x\) likewise orthogonalized with respect to \(span\{u_{n_1}, ..., u_{n_{k-1}}\}\)) such that its contribution \(-(u_{n_k}^T x_{res})^2\) in (6) decreases. This leads to the Hebbian-like update rule

\[
y \leftarrow u_{n_k}^T x_{res}
\]

\[
\Delta u_{n_k} \leftarrow \epsilon_t \cdot y \cdot x_{res} .
\] (7)

The updated basis vector \(u_{n_k}\) is normalized to unit length. Subsequently, \(x_{res}\) is orthogonalized with respect to \(u_{n_k}\), thus becoming the residual vector for the update of \(u_{n_{k+1}}\).

How many basis vectors are updated with the learning rule depends on the sparsity parameter \(K\). Please note, however, that due to the required orthogonalization, all basis vectors are modified even if only \(K\) were updated by (8). A learning step is completed, when the last basis vector \(u_{n_N}\) has been normalized. With this scheme of iterative Gram-Schmidt orthogonalization, \(U\) is most certainly an orthogonal basis. The learning rate \(\epsilon_t\) cools down with the number of conducted learning steps. Algorithm 1 lists OSC in pseudo code.

\begin{algorithm}

\textbf{Input:} training data set \(X = (x_1, ..., x_L) \in \mathbb{R}^{N \times L}\) number of learning steps \(t_{max}\) initial and final learning rate \(\epsilon_{init} \geq \epsilon_{final}\) sparsity level \(K\) (default: \(K = N\))

\textbf{Output:} orthogonal basis \(U = (u_1, ..., u_N) \in \mathbb{R}^{N \times N}\)

1: Start with random orthogonal basis \(U\)
2: for all \(t = 0, ..., t_{max}\) do
3: \(\epsilon_t \leftarrow \epsilon_{init} \cdot \epsilon_{final}^{1/t_{max}}\)
4: \(\text{select sample } x \text{ from } X \text{ randomly and set } x_{res} \leftarrow x\)
5: \(\text{determine a sequence } n_1, ..., n_N, \text{ such that } (u_{n_1}^T x)^2 \geq ... \geq (u_{n_N}^T x)^2 \) holds
6: for all \(k = 1, ..., N\) do
7: \(\text{for all } l = 1, ..., (k-1) \) do
8: \(u_{n_l} \leftarrow u_{n_l} - (u_{n_k}^T u_{n_l}) u_{n_l}\)
9: end for
10: if \(k \leq K\) then
11: \(y \leftarrow u_{n_k}^T x_{res}\)
12: \(u_{n_k} \leftarrow u_{n_k} + \epsilon_t \cdot y \cdot x_{res}\)
13: end if
14: \(\text{normalize } u_{n_k} \text{ to unit length}\)
15: \(x_{res} \leftarrow x_{res} - (u_{n_k}^T x_{res}) u_{n_k}\)
16: end for

end for

\end{algorithm}

The OSC algorithm starts with a randomly initialized basis \(U\). For each learning step an entire update of \(U\) takes place. At the beginning of each learning step \(t\), the current learning rate \(\epsilon_t\) is set and a training sample \(x\) is randomly selected from the given training data set \(X\). First, OSC determines an index sequence \(n_1, ..., n_N\) that sorts the squared coefficients of \(x\) in the current basis \(U\), s.t. \((u_{n_1}^T x)^2 \geq (u_{n_2}^T x)^2 \geq ... \geq (u_{n_N}^T x)^2\). Note that the contribution of the selected sample \(x\) to the cost function (5) is given by

\[
E_K(x, U) = -x^T U D_K(x, U) U^T x = -\sum_{k=1}^{K} (u_{n_k}^T x)^2 .
\] (6)

As a consequence, costs are reduced if the sum of the \(K\) largest squared coefficients is increased. Loosely speaking, this can be interpreted as concentrating the energy distribution of the sparse coefficients of \(x\). The determined index sequence defines the order of basis vector updates, starting with the basis vector which contributes most to (6). Before a basis vector \(u_{n_{k}}\) is updated, it is orthogonalized with respect to \(span\{u_{n_1}, ..., u_{n_{k-1}}\}\), the span of basis vectors that were already updated during the current learning step. Then, the orthogonalized basis vector \(u_{n_k}\) is updated by gradient descent depending on the residual vector \(x_{res}\) (the original training sample \(x\) likewise orthogonalized with respect to \(span\{u_{n_1}, ..., u_{n_{k-1}}\}\)) such that its contribution \(-(u_{n_k}^T x_{res})^2\) in (6) decreases. This leads to the Hebbian-like update rule

\[
y \leftarrow u_{n_k}^T x_{res}
\]

\[
\Delta u_{n_k} \leftarrow \epsilon_t \cdot y \cdot x_{res} .
\] (7)

The updated basis vector \(u_{n_k}\) is normalized to unit length. Subsequently, \(x_{res}\) is orthogonalized with respect to \(u_{n_k}\), thus becoming the residual vector for the update of \(u_{n_{k+1}}\).

How many basis vectors are updated with the learning rule depends on the sparsity parameter \(K\). Please note, however, that due to the required orthogonalization, all basis vectors are modified even if only \(K\) were updated by (8). A learning step is completed, when the last basis vector \(u_{n_N}\) has been normalized. With this scheme of iterative Gram-Schmidt orthogonalization, \(U\) is most certainly an orthogonal basis. The learning rate \(\epsilon_t\) cools down with the number of conducted learning steps. Algorithm 1 lists OSC in pseudo code.

C. Computational Complexity of OSC

For a single OSC learning step, drawing a training sample \(x\), setting residual vector \(x_{res}\) (line 4) and sorting the coefficients (line 5) has complexity \(O(N^2) + O(N \log N)\). The loop in lines 6-16 iterates \(O(N)\) times over all basis vectors \(u_{n_k}\). The Gram-Schmidt steps for each \(u_{n_k}\) (lines 7-9) have at most a complexity of \(O(N^2)\). A single Hebbian-like update (lines 11-12) of a \(u_{n_k}\) has a complexity of \(O(N)\). The length normalization of \(u_{n_k}\) (line 14) and the update of \(x_{res}\) (line 15) take likewise \(O(N)\). Putting all together and taking the outer loop over the learning steps into account gives a computational complexity of

\[
O \left( t_{max} \left( N \log N + KN + N^2 + N^3 \right) \right) ,
\] (9)

which is bounded by \(O(t_{max}N^3)\).

---

\(^1\)A first sketch of the OSC algorithm and preliminary results have been presented at the workshop New Challenges in Neural Computation 2013 [38].
D. Convergence Properties of OSC

OSC is an online learning algorithm that updates the sparse coding basis \( U \) for each presented training sample \( x \). In the following we provide a theorem which assures that a learning step by OSC increases the sparsity of the representation of the given \( x \), i.e. decreases the cost contribution of this sample to the overall cost function. Hence, OSC performs stochastic descent on the cost function which has to be minimized. For small enough step sizes \( \varepsilon \) this can be proven, but seems to be valid also for large \( \varepsilon \) according to our numerical experiments.

Theorem 1: Given an orthogonal basis \( U \). If \( \varepsilon > 0 \) is small enough, applying an OSC learning step to an arbitrary non-zero \( x \) yields a new orthogonal basis \( U' \) such that for the sequences \( (u_{n1}^T x)^2 \geq (u_{n2}^T x)^2 \geq \ldots \geq (u_{n1}^T x)^2 \) and \( (u_{n1}^T x)^2 \geq (u_{n2}^T x)^2 \geq \ldots \geq (u_{nN}^T x)^2 \) the ordering

\[
\frac{(u_{n_{k+1}}^T x)^2}{(u_{n_k}^T x)^2} \leq \frac{(u_{n_{k+1}}^T x)^2}{(u_{n_k}^T x)^2} \quad (10)
\]

holds for all \( k = 1, \ldots, N - 1 \).

Theorem 1 states that by an OSC learning step the magnitude of a coefficient decreases relative to its predecessor in the sequence of sorted coefficients. This means, that after an OSC learning step the squared coefficients obey a stronger decay. Figure 1 illustrates, from an experiment with natural image patches, the sorted squared coefficients before and after an OSC learning step. Clearly, after the learning step more energy is distributed over less coefficients. However, the total amount of energy is preserved, because \( U \) is an orthogonal basis. Hence, the sparsity of the encoding of the given \( x \) is increased.

This means that an OSC learning step reduces the costs (6) with respect to the presented training sample. It might not be obvious that OSC minimizes cost function (5) with respect to the entire training data set. However, a stochastic descent of (5) is due to the pattern-by-pattern scheme of OSC similar to a stochastic gradient descent. From the experiment with natural image patches described in Subsection III-B, Figure 2 illustrates the temporal course of the total costs (5) for the OSC basis \( U \) over a full learning phase. It can be seen that OSC reduces the total costs (5).

E. K-OSC and "full" OSC

In general, \( E_{X,K}(U) \) has different minima for different \( K \) and OSC will provide different solutions \( U_{K}^{*} \). However, there are situations where it is suitable to choose user parameter \( K \) of the OSC algorithm as \( K = N \) and take the output of this "full" OSC as a kind of universal solution \( U^{*} \) which minimizes (5) for many different \( K \). In these cases it is not necessary to learn an individual basis for each individual \( K \).

Such a situation is given, for example, if \( U_{K}^{*} \) does not change although \( K \) is further increased. Starting with cost function (5) and some rearrangements we can write

\[
E_{X,K+1}(U) = - \sum_{n=1}^{N} \sum_{k=1}^{K+1} \sum_{x \in S_{n}^{U}(U)} (u_{n}^{T} x)^2
\]

\[
= E_{X,K}(U) - \sum_{n=1}^{N} \sum_{x \in S_{n}^{U_{K}^{*}}(U)} (u_{n}^{T} x)^2 \quad (11)
\]

\( S_{n}^{U}(U) \) is the set of those \( x \) for which \( (u_{n}^{T} x)^2 \) is the \( k \)-th largest term in the sequence \( (u_{n}^{T} x)^2 \geq (u_{n}^{T} x)^2 \geq \ldots \geq (u_{n}^{T} x)^2 \). Since \( U_{K}^{*} \) is a minimum of the first term in (11), a sufficient condition for \( U_{K}^{*} \) to remain a minimum also of \( E_{X,K+1} \) is \( U_{K}^{*} \) to be a minimum also of the second term in (11). This is the case if the gradient of the second term with respect to each \( u_{n} \) vanishes at \( U = U_{K}^{*} \), i.e.,

\[
\sum_{x \in S_{n}^{U_{K}^{*}}(U)} u_{n}^{T} x = 0 \quad \forall \ n \in \{1, \ldots, N\}, \quad (12)
\]
Condition (12) holds, for example, if the \( x \) are \( K \)-sparse in \( U^*_K \), since each element in the sum of the gradient is zero. This is still valid if the signals are \( K \)-sparse in \( U^*_K \) except for additive isotropic noise, since for symmetry reasons the contributions in the sum cancel out. Hence, if we have \( K \)-sparse signals without or with isotropic noise, working with the "full" version of OSC, i.e. with \( U^* \), gives us the correct solution, even if we do not know the given sparsity level \( K \).

Another scenario appropriate for using \( U^* \) is given if each signal is a linear superposition of orthogonal atoms, which are drawn randomly and independently (without replacement) from the dictionary \( V = (v_1, ..., v_N) \). The coefficients are drawn independently from the probability distribution \( P_1 \) for the first chosen atom, from \( P_2 \) for the second chosen atom, etc. Because of the independencies we have \( V = U^*_1 = U^*_2 = ... = U^*_{N-1} = U^* \). Again, it is convenient to use the "full" OSC (\( K = N \)), instead of having to decide how to choose the parameter \( K \).

There are, of course, further scenarios where \( U^* \) is either the exact or at least a good approximative solution for many different \( K \) values. Indeed, in our experiments we have consistently observed this convenient property (see Subsections III-A, III-B, and III-C). Obviously, an algorithm that does not rely on knowing the sparsity level \( K \) is more practical.

## III. Results

In the following, we label results obtained by the “full” OSC variant (\( K = N \)) simply as “OSC” and use otherwise the annotation “K-OSC”.

### A. Results on Synthetic Data

We investigated how well K-SVD, the approach of Mishali et al. ([31]), CA, K-OSC, and OSC can recover a generating orthogonal basis from \( K \)-sparse synthetic data. Note, that K-SVD is an algorithm for finding arbitrary, non-orthogonal sparse coding dictionaries and does therefore not benefit from the orthogonality of an underlying dictionary. Nevertheless, orthogonality is a good-natured scenario for K-SVD, because the mutual coherence is minimal.

To generate a synthetic data set, we fixed signal dimensionality to \( N = 256 \) and sample size to \( L = 1000 \), whereas the sparsity level \( K \in \{2, 6, ..., 58, 62\} \) was gradually varied. Each data sample was generated as \( 16 \times 16 \) patch being \( K \)-sparse in the non-standard 2D Haar wavelet basis. The support pattern of each sample, i.e., the \( K \) locations of non-zero coefficients in the Haar wavelet domain were uniformly selected at random. Then, the non-zero coefficients were drawn randomly from a standard Gaussian distribution. In order to investigate deviations of recovery rates over multiple runs, we created 10 data sets for each sparsity level.

We applied the five algorithms to the generated data sets and provided all but OSC with the known \( K \) as user parameter. Each method conducted 100 learning epochs. To measure recovery performance for each run we followed a procedure similar to the one used in [21]. We first determined the “best matching pairs” between estimated and generating basis vectors. This was done by first sorting the overlaps\(^2\) of all \( N^2 \) possible basis vector pairs in decreasing order. Subsequently, the “best matching pairs” were assigned according to that sequence with the requirement that estimated basis vectors and generating basis vectors obey a one-to-one assignment. We considered a generating basis vector as recovered, if it has an overlap of at least 0.8 to its matched estimated version. The recovery rate of a full basis is expressed as the ratio of recovered basis vectors.

This is the author’s version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication. The final version of record is available at http://dx.doi.org/10.1109/TCI.2016.2557065

![Figure 3. Mean recovery rates and standard deviations for synthetic data sets (\( L = 1000 \) patches of \( 16 \times 16 \) being \( K \)-sparse in the 2D Haar wavelet basis).](image-url)
However, with increasing $K$ the recovery performance of OSC and K-OSC decays the slowest.

**B. Results on Natural Image Patches**

We let OSC learn orthogonal bases to sparsely encode natural image patches. We extracted image patches from set one of the Nature Scene Collection [39], i.e., from images of nature scenes containing no man made objects or people. The uncompressed RGB images have a resolution of $2844 \times 4284$ pixels. The color channels are linearly scaled, each with a depth of 16 bits per pixel (bpp). To each color channel the logarithm to the base of 2 and a subsequent scaling into the double precision floating point range $[0, 1]$ was applied. Subsequently, the color images were converted to grayscale images. From the entire set of 308 images, we randomly selected 250 images for learning. From each image, we extracted 400 patches of size $16 \times 16$ pixels at random positions. These 105,000 image patches were used for training. In the same manner, a test data set with 23200 patches was generated from the remaining 58 images. Data preprocessing comprised the sample-wise subtraction of the DC component and of the mean vector.

We conducted $t_{\text{max}} = 10^7$ learning steps with OSC. The initial and final learning rates were manually set to $\varepsilon_{\text{init}} = 10$ and $\varepsilon_{\text{final}} = 10^{-2}$ without an extensive parameter validation.

Figure 6 illustrates the orthogonal basis learned by OSC on natural image patches. The OSC basis resembles a wavelet decomposition. On different scales, the basis patches show selectivity for inputs with particular frequencies, orientations, and spatial localizations.

For the test data set, Figure 7 illustrates the average optimal $K$-term approximation performance of different bases measured by the signal-to-noise-ratio (SNR). On the one hand, we compare OSC to the predefined orthogonal transform bases of 2D DCT and non-standard 2D Haar wavelets which are known to provide decent sparse representations. On the other hand, we compare OSC to bases learned by PCA$^3$, CA, and K-SVD$^4$.

3With PCA, $K$-term approximations are derived from the $K$ first PCs
4With K-SVD, $K$-term approximations are obtained by Batch OMP [40]
We transformed the images to the double precision floating point range $[0, 1]$ and resized them to $16 \times 16$ pixels using bicubic interpolation. From each sample its DC component was subtracted. We conducted $t_{\text{max}} = 3.6 \cdot 10^5$ learning steps with OSC. The initial and final learning rates were set to $\varepsilon_{\text{init}} = 2.8 \cdot 10^{-3}$ and $\varepsilon_{\text{final}} = 2.8 \cdot 10^{-3}$, respectively.

Figure 8 depicts the learned OSC basis patches. Note that many OSC basis patches learned on the MNIST training set show sensitivity for particular digits or digit combinations. Furthermore, some basis patches show sensitivity to localized grating patterns with different orientations. Note, that some of these grating patterns have curved shapes.

In Figure 9, we again compare the average optimal $K$-term approximation performance of OSC, K-OSC, 2D DCT, 2D Haar wavelets, PCA, CA, and K-SVD, this time on the MNIST test data set. K-SVD has a slightly better performance for small $K \leq 32$. For the remaining sparsity levels OSC and K-OSC have a clearly superior $K$-term approximation performance compared to the alternative methods. In order to achieve an average reconstruction performance of, e.g. 40 dB, OSC uses on average approximately 10% less non-zero coefficients than the second best performing Haar wavelet basis. Note that the single OSC basis yields, again, nearly the same $K$-term approximation performance as the K-OSC bases to which the sparsity level of the reconstruction was provided.

C. Results on Images of Handwritten Digits

The same experiment was conducted with the MNIST data set [41], i.e., with images of handwritten digits. The training data set comprises $6 \cdot 10^4$ and the test data set $10^4$ grayscale images of size $28 \times 28$ with a gray-level depth of 8 bit. We transformed the images to the double precision floating point range.

IV. APPLICATIONS

A. Image Compression

To demonstrate the applicability of OSC, we conducted image compression experiments with gray level images (8 bit

Note that by K-SVD non-orthogonal complete dictionaries were learned. As for OSC and K-OSC we let CA and K-SVD learn for 100 epochs. Note that a comparison to the approach of Mishali et al. ([31]) was not possible, because its support recovery stage requires the existence of a strictly $K$-sparse representation and returned, in our attempts, maximally non-sparse support patterns for the training data.

For $K \leq 32$, K-SVD performs slightly better than OSC, K-OSC, DCT, and CA which have nearly equal approximation performances. For larger $K$, OSC and K-OSC equally perform best at the task to sparsely encode the test data. Both achieve a slightly better encoding performance compared to the DCT whereas the superiority compared to Haar wavelets, PCA, CA, and K-SVD is more striking. It is remarkable that the single OSC basis (see Figure 6) yields the same $K$-term approximation performance as the K-OSC bases which were individually learned for each tested sparsity level. It appears that this universality property of OSC is not only limited to particular artificial data settings, but also holds for natural image data.

We also investigated the stability of OSC by fixing the training set as well as the learning parameters and applied the algorithm repeatedly with different initial random bases and different random training sequences. On the fixed test data set, we computed the average optimal $K$-term approximation performance for each OSC basis learned in our 20 runs. We found, that the standard deviation of the $K$-term approximation performance was less than 0.081 dB for any $K \leq 254$, and even less than 0.017 dB for any $K \leq 230$ (i.e. $K/N \leq 0.9$).
of each OSC basis vector appeared among the 8 OSC basis, therefore, it was counted 8 × frequency (and decreasing relevance). For the 8 DCT has been replaced by an 8 × 8 OSC [6], where the 8 × 8 OSC basis which was learned as described in Subsection III-B.

We derived an OSC codec from the JPEG baseline codec [6], where the 8 × 8 DCT was replaced by an 8 × 8 OSC basis. We used the Independent JPEG Groups JPEG library. The JPEG codec was done by applying the program pnmtojpeg, see Appendix for the rate distortion analysis for the remaining test images).

Nevertheless, at low bitrates the JPEG2000 codec is still superior. Note, however, that it benefits from the multi-scale representation of images in the wavelet domain. Both, the JPEG and OSC codecs are patch based and suffer from blocking artifacts at very low bitrates. This might be one reason for their limitations compared to JPEG2000.

B. Image Denoising

We conducted experiments to assess the applicability of OSC to image denoising. The dictionaries were learned on the training data set described in Subsection III-B. Following the denoising framework proposed in [44], we distorted 9 gray value test images (512 × 512 pixels) with additive Gaussian noise with standard deviations \( \sigma \in \{2, 5, 10, 15, 20, 25, 50\} \). After the noise was added, the noisy images were clipped to the range [0, 255]. From a noisy image, patches (16 × 16 pixels) were extracted from all locations and sparsely approximated by OMP applying a regularization with respect to the full size image reconstruction error. Note that a sparse approximation computed by OMP is optimal if the dictionary is orthogonal, i.e., the most relevant coefficients are retained. The denoised image is constructed by fusing the sparsely approximated patches. The gray value of each pixel is averaged from all its overlapping patches. For the entire image denoising procedure we used the ompdenoise2.m function of the KSVDBox v13 in combination with OMPBox v10 [40] with parameters as proposed in [44].

We compared image denoising performance between dictionaries learned by K-SVD, CA and OSC for 100 epochs. For
IEEE TRANSACTIONS ON COMPUTATIONAL IMAGING, VOL. XX, NO. XX, XX 20XX

Table I

<table>
<thead>
<tr>
<th>Image</th>
<th>Lena (512 × 512)</th>
<th>Peppers (512 × 512)</th>
<th>Pirate (512 × 512)</th>
<th>Barbara (512 × 512)</th>
<th>Boat (512 × 512)</th>
<th>Finger (512 × 512)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-SVD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K-SVD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K-SVD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table I: Image denoising based on sparse approximations of 16 × 16 image patches. Denoising performance is measured in terms of PSNR (dB) between original image and denoised estimate.

CA, OSC, K-SVD, and OSC dictionary perform comparably well depending on the combination of image and noise level is highlighted in the table. The denoising performance is listed in Table I in terms of PSNR (dB) averaged over 5 runs. The largest PSNR for each combination of image and noise level is highlighted in bold face. The experiments show that the K-SVD, CA, and OSC dictionary perform comparably well depending on the chosen images and noise levels. However, for K-SVD and CA, performance depends on the chosen parameters codebook size (only for K-SVD) and sparsity level $K$, while OSC does not require the optimization of such parameters.

V. DISCUSSION AND CONCLUSION

In this paper, we addressed the problem of learning orthogonal bases for sparse coding.

Inspired by traditional sparse coding algorithms, we first proposed the Canonical Approach (CA) which alternates between the adaptation of the basis and the update of the sparse coefficients. We showed that CA yields high performance at recovering a generating orthogonal basis from synthetic data, if the sparsity of the data is known and rather high.

As the main part of the paper, we presented Orthogonal Sparse Coding (OSC), an unsupervised online learning algorithm, which is based on Hebbian learning and iterative Gram-Schmidt orthogonalization. OSC is able to identify the generating orthogonal basis from synthetic data even if the sparsity of the data is unknown or low. In contrast, K-SVD, CA, and the approach of Mishali et al. ([31]) had difficulties to converge at the basis identification task, particularly for the more challenging settings.

On natural image patches, OSC learns a basis that resembles wavelets with sensitivity to particular frequencies, orientations, and spatial localizations. In terms of optimal $K$-term approximation performance, OSC performs clearly better than the 2D Haar basis, PCA, CA, and K-SVD (except for very small $K$) and slightly better than the 2D DCT basis.

On images of handwritten digits, OSC learns a basis of patches that are sensitive to certain digits or combinations of digits, because they have adapted to the specific image class. Furthermore, the basis patches show localized gratings of various shapes. Again, the average optimal $K$-term reconstruction performance of the learned OSC basis is better than alternative predefined and learned bases.

For both, the synthetic and the real world data sets, we found that learning a single basis with OSC (setting user parameter $K = N$) provides a “universal” solution which gives equally good optimal $K$-term approximations for various sparsity levels $K$ as compared to learning specific bases with the K-OSC algorithm for each $K$ individually. This is very advantageous, since the “true” sparsity level $K$ is often unknown in practice.

In contrast to PCA, encoding and dimensionality reduction are not based on a fixed linear subspace, but the best encoding subspace is selected individually for each data sample. Our results show that this additional freedom of choice is beneficial in the case of natural images and even more so for more specific signals like handwritten digits.

Moreover, we demonstrated the applicability of OSC for image compression by showing that OSC-based compression yields a better rate-distortion performance than JPEG, although it remains inferior to JPEG2000. We also demonstrated that orthogonal dictionaries learned by CA and OSC are useful for image denoising. CA and OSC dictionaries achieve denoising performance comparable to redundant dictionaries learned by K-SVD.

Regarding possible extensions, one should note that OSC is a flexible algorithm which can also learn undercomplete orthogonal dictionaries, i.e., orthogonal $M$-frames with $M < N$. With such an additional user-defined parameter $M$ for the dictionary size, the orthogonal $M$-frame $U \in \mathbb{R}^{N \times M}$ would be updated as in Algorithm 1 except for lines 6 and 7, where $N$ would be replaced by $M$. A further extension could be to learn multiple orthogonal bases from only one training set. In this case, OSC would update, for each data point, the best sparse coding basis only. Finally, one could use OSC for compressed sensing and adaptive hierarchical sensing as outlined in [45].
ACKNOWLEDGMENT

The research is funded by the DFG Priority Programme SPP 1527, grant number MA 2401/2-1.

REFERENCES


APPENDIX

PROOF OF THEOREM 1

Proof: In the following, we use the assumption that $\varepsilon$ is small. We develop all expressions up to first order in $\varepsilon$ and treat terms of order $\varepsilon^2$ and higher as vanishing.

Without any loss of generality, we assume $(u_k^T x)^2 \geq (u_1^T x)^2 \geq \ldots \geq (u_{\ell}^T x)^2$ which defines the order of basis vector updates. Each $u_k$, except for $u_1$, is updated in two
steps. First, the Gram-Schmidt orthogonalization
\[ v_k = u_k - \sum_{l=1}^{k-1} (u_l^T u_k)u_l' \]  
followed by the normalized Hebbian-like main update
\[ u_k' = \frac{v_k + \varepsilon (v_k^T x_k)x_k}{\|v_k + \varepsilon (v_k^T x_k)x_k\|_2} \]  
where
\[ x_k = x - \sum_{l=1}^{k-1} (u_l^T x)u_l' \]  
\[ u_1 \] is only updated by (14) due to (13). In that sense \( v_1 = u_1 \) and \( x_1 = x \) due to (15).

We will show by induction that
\[ v_k = u_k - \varepsilon (u_k^T x) \sum_{l=1}^{k-1} (u_l^T x)u_l' + O(\varepsilon^2) \]  
Note that by (16) it holds \( \|v_k\|_2 = 1 + O(\varepsilon^2) \). Hence, the Taylor expansion of update step (14) up to first order in \( \varepsilon \) is
\[ u_k' = v_k + \varepsilon (v_k^T x)(x_k - (v_k^T x)v_k) + O(\varepsilon^2) \]  
Note that (14) is a Oja learning rule, i.e., a Hebbian learning rule with a normalization constraint. We apply the same expansion as in Section 4 of [46].

Furthermore, since \( v_k^T x_k = v_k^T x \) and with (16) we have \( v_k^T x_k = u_k^T x + O(\varepsilon) \) as well as \( (v_k^T x_k)v_k = (u_k^T x)u_k + O(\varepsilon) \). Hence, (17) can be restated as
\[ u_k' = v_k + \varepsilon (u_k^T x) (x_k - (u_k^T x) u_k) + O(\varepsilon^2) \]  
We will now show (16) by induction.

**Initial Step** \( k = 1 \). According to (13), we have by definition \( v_1 = u_1 \) which satisfies (16).

**Induction Step** \( (k-1) \rightarrow k \). According to (13), we have by definition
\[ v_k = u_k - \sum_{l=1}^{k-1} (u_l^T u_k)u_l' \]  
Due to induction hypothesis (16), \( u_k' \) can be restated according to (18). In addition to (16) we will use \( u_l^T u_k = 0 \) and \( v_l^T u_k = O(\varepsilon^2) \) as well as \( u_k^T x_1 = u_k^T x + O(\varepsilon) \).

\[ v_k = \sum_{l=1}^{k-1} \left[ (v_l + \varepsilon (u_l^T x)(x_l - (u_l^T x) u_l))u_l' \right] + O(\varepsilon^2) \]  
\[ = u_k - \varepsilon (u_k^T x) \sum_{l=1}^{k-1} (u_l^T x) u_l' + O(\varepsilon^2) \]  
\[ = u_k - \varepsilon (u_k^T x) \sum_{l=1}^{k-1} (u_l^T x)(v_l + \varepsilon (u_l^T x)(x_l - (u_l^T x) u_l)) + O(\varepsilon^2) \]  
\[ = u_k - \varepsilon (u_k^T x) \sum_{l=1}^{k-1} (u_l^T x) u_l + O(\varepsilon^2) \]  
The induction is complete.

Combining (18) and (16) gives us up to first order in \( \varepsilon \)
\[ u_k' = u_k + \varepsilon (u_k^T x) \left( x_k - \sum_{l=1}^{k-1} (u_l^T x) u_l \right) \]  
Hence, for small \( \varepsilon \) and with (15) we obtain
\[ \frac{(u_{k+1}^T x)^2}{(u_k^T x)^2} = \left( \frac{1 + \varepsilon}{1 + \varepsilon} \right)^2 \left( 1 + \varepsilon \left| x_k^T x - \sum_{l=1}^{k-1} (u_l^T x)^2 \right| \right)^2 \]  
\[ \leq \frac{(u_{k+1}^T x)^2}{(u_k^T x)^2} \]  
since the square bracket in the nominator is smaller than the square bracket in the denominator.

**Zoom In of Compressed Image Cathedral**

![Figure 11](image-url)  
An image region (120 x 120 pixels) cropped from the upper left part of test image Cathedral after compression by the JPEG2000, OSC, and JPEG codecs at compression rate 0.29 bpp.
Henry Schütze studied Computer Science at the Brandenburg University of Technology, Cottbus, Germany and at the University of Lübeck, Germany. He graduated in 2011 with an MSc and is now research assistant at the Institute for Neuro- and Bioinformatics of the University of Lübeck, where he pursues a PhD degree. His main research interests include Sparse Coding and Compressed Sensing.

Erhardt Barth received the PhD degree in electrical and communications engineering from the Technical University of Munich, Germany. He is a Professor at the Institute for Neuro- and Bioinformatics, University of Lübeck, Germany, where he leads the research on human and machine vision. He has conducted research at the Universities of Melbourne and Munich, the Institute for Advanced Study in Berlin, and the NASA Vision Science and Technology Group in California.

Thomas Martinetz is full professor of computer science and director of the Institute for Neuro- and Bioinformatics at the University of Lübeck, Germany. He studied Physics at the Technical University of Munich, Germany and did his PhD work in Biophysics at the Beckman Institute for Advanced Science and Technology of the University of Illinois at Urbana-Champaign, USA. From 1991 to 1996 he led the project Neural Networks for automation control at the Corporate Research Laboratories of the Siemens AG in Munich. From 1996 to 1999 he was Professor for Neural Computation at the Ruhr-University of Bochum and head of the Center for Neuroinformatics.