SPARSE NONNEGATIVE MATRIX FACTORIZATION USING ℓ^0-CONSTRAINTS

Robert Peharz, Michael Stark, Franz Pernkopf

Signal Processing and Speech Communication Lab
University of Technology, Graz

ABSTRACT

Although nonnegative matrix factorization (NMF) favors a part-based and sparse representation of its input, there is no guarantee for this behavior. Several extensions to NMF have been proposed in order to introduce sparseness via the ℓ^1-norm, while little work is done using the more natural sparseness measure, the ℓ^0-pseudo-norm. In this work we propose two NMF algorithms with ℓ^0-sparseness constraints on the bases and the coefficient matrices, respectively. We show that classic NMF [1] is a suited tool for ℓ^0-sparse NMF algorithms, due to a property we call *sparseness maintenance*. We apply our algorithms to synthetic and real-world data and compare our results to sparse NMF [2] and nonnegative K-SVD [3].

1. INTRODUCTION

Nonnegative Matrix Factorization (NMF) aims to factorize a nonnegative matrix $\mathbf{X}$ into a product of nonnegative matrices $\mathbf{W}$ and $\mathbf{H}$, i.e. $\mathbf{X} \approx \mathbf{W}\,\mathbf{H}$. Therefore, the task is to find nonnegative matrices $\mathbf{W}$ and $\mathbf{H}$ such that an error measure, like the Frobenius norm $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2$ or the Kullback-Leibler divergence $D(\mathbf{X}||\mathbf{W}\,\mathbf{H})$ is minimized. When we assume the columns of $\mathbf{X}$ as multidimensional observations of some process, the columns of $\mathbf{W}$ can be interpreted as basis vectors, whereas the rows of $\mathbf{H}$ contain the corresponding coding coefficients. Application domains of NMF are manifold, such as data compression and data analysis, feature extraction [4], denoising [5], and others. Lee and Seung [4] noted that NMF favors a sparse and localized representation of its input, in contrast to other linear matrix decompositions such as principal component analysis and k-means clustering.

Other authors noted that nonnegativity constraints alone do not guarantee a sparse representation of the input, and further, that the degree of sparseness cannot be controlled. Therefore, various extensions have been proposed in order to incorporate sparseness constraints in NMF. Hoyer [6] proposed an algorithm to minimize the objective $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2 + \lambda \sum_{ij} |H|_{ij}$, which penalizes the ℓ^1-norm of the coefficients matrix $\mathbf{H}$. Eggert and Koerner [7] used the same objective, but proposed an alternative update which implicitly normalizes the columns of $\mathbf{W}$ to unit length. Further, Hoyer [2] defined a sparseness function of an arbitrary vector via the ℓ^1-norm and presented an NMF algorithm which constrains the columns of $\mathbf{W}$ or $\mathbf{H}$ to a given sparseness value. Two other extensions of classic NMF which aim to achieve a part-based and sparse representation are local NMF [8] and nonsmooth NMF [9].

While most of these extensions of NMF introduce sparseness via constraining or penalizing the ℓ^1-norm, little to no work is concerned about NMF with ℓ^0-sparseness constraints. Although the ℓ^0-pseudo-norm is the most natural and intuitive sparseness measure, there is a good reason to define sparseness via the ℓ^1-norm: In contrast to the ℓ^0-norm, the ℓ^1-norm is convex, and it is known that a solution for a ℓ^1-constrained problem approximates the solution for the ℓ^0-constrained problem [10]. However, there are at least two reasons which justify an ℓ^0-constrained version of NMF. Firstly, such a method, although approximate, is useful, since we are able to constrain the bases or coefficient vectors to have exactly the desired number of nonzero entries. Secondly, the joint optimization of $\mathbf{W}$ and $\mathbf{H}$ is a non-convex problem per se, which means that all NMF algorithms proposed so far converge to a local minimum only. Therefore also ℓ^1-constrained NMF methods are only approximative solutions.

In this paper we present two NMF algorithms with ℓ^0-constrains on the columns of $\mathbf{H}$ and $\mathbf{W}$, respectively. To constrain the coefficient matrix $\mathbf{H}$ we follow the framework of nonnegative K-SVD [3], which itself can be seen as an NMF algorithm with ℓ^0-constraints on $\mathbf{H}$. To constrain the bases matrix $\mathbf{W}$ we proceed similar as in the work of Hoyer [2]. Generally, we show that the standard NMF update rules [1] are suited for ℓ^0-sparse NMF, due to a property which we call sparseness maintenance. Throughout this paper, $\mathbf{X}$ denotes a $D \times N$ data matrix, $\mathbf{W}$ denotes a $D \times K$ bases matrix (or dictionary) and $\mathbf{H}$ denotes a $K \times N$ coefficient matrix.

The paper is organized as follows: In Section 2 we review NMF [1] and sparse NMF [2]. Further, after reviewing the sparse coding problem, we discuss the K-SVD algorithm by Aharon et al. [11] and its nonnegative variant [3]. In Section 3 we introduce our algorithms for ℓ^0-sparse NMF. We report experimental results on synthetic and real-world data in Section 4. Finally, Section 5 concludes this paper.

Acknowledgement: This work was supported by the Austrian Science Fund (Project numbers S10604-N13 and P22488-N23)

2. RELATED WORK

2.1. Nonnegative Matrix Factorization

Lee and Seung [1] showed that the multiplicative update rules

$$\mathbf{H} \leftarrow \mathbf{H} \otimes \frac{(\mathbf{W}^T\mathbf{X})}{(\mathbf{W}^T\mathbf{W}\,\mathbf{H})}, \qquad (1)$$

$$\mathbf{W} \leftarrow \mathbf{W} \otimes \frac{(\mathbf{X}\,\mathbf{H}^T)}{(\mathbf{W}\,\mathbf{H}\,\mathbf{H}^T)}, \qquad (2)$$

converge to a local minimum of $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2$, where $\otimes$ and $\frac{\cdots}{\cdots}$ denote element-wise multiplication and division, respectively. Nonnegativity of $\mathbf{W}$ and $\mathbf{H}$ is maintained, since the updates consist of products of nonnegative factors only. They also provided update rules for the Kullback-Leibler divergence [1].

2.2. Sparse NMF

Hoyer [2] provided an NMF method with sparseness constraints on the columns of $\mathbf{W}$, the columns of $\mathbf{H}$, or both. In this work, the sparseness of an arbitrary D-dimensional vector $\mathbf{x}$ is defined as:

$$\mathrm{sparseness}(\mathbf{x}) = \frac{\sqrt{D} - L_1(\mathbf{x})/L_2(\mathbf{x})}{\sqrt{D} - 1}, \qquad (3)$$

where $L_1(\mathbf{x})$ and $L_2(\mathbf{x})$ denote the ℓ^1- and ℓ^2-norms, respectively: $L_1(\mathbf{x}) = \sum_{i=1}^{D} |x_i|$, $L_2(\mathbf{x}) = \sqrt{\sum_{i=1}^{D} x_i^2}$. Indeed, $\mathrm{sparseness}(\mathbf{x})$ is 0 if all entries of $\mathbf{x}$ are nonzero and of the same absolute value, and 1 for the sparsest possible vector, i.e. when only one entry is nonzero. For all other $\mathbf{x}$, the function smoothly interpolates between these extreme cases.

The sparse NMF algorithm performs gradient descend on the cost function $\|\mathbf{X} - \mathbf{W}\mathbf{H}\|_F^2$ and projects the columns of $\mathbf{W}$ or $\mathbf{H}$ (or both) onto the set of element-wise nonnegative vectors with desired sparsity according to Eq. (3) after each iteration. We refer to this algorithm as ℓ^1-sparse NMF for the remainder of this paper.

2.3. Sparse Coding

A sparse coder aims to approximate a vector $\mathbf{x}$ using a linear combination of maximal L bases (which are called atoms in this context): $\mathbf{x} \approx \sum_{i=1}^{L} h_{z_i}\,\mathbf{w}_{z_i}$. Here $\mathbf{z} = (z_1, \dots, z_L)^T$ is an index vector which holds the indices of the selected atoms, $\mathbf{w}_{z_i}$ denotes the $z_i{}^{\text{th}}$ column of $\mathbf{W}$, and $(h_{z_1}, \dots, h_{z_L})^T$ are the weighting coefficients. Usually we have $L \ll K$, hence the term *sparse* coding. We can define the sparse coding problem as minimization of $\|\mathbf{x} - \mathbf{W}\,\mathbf{h}\|^2$, s.t. $L_0(\mathbf{h}) \leq L$, where $L_0(\cdot)$ denotes the ℓ^0-pseudo-norm, i.e. the number of nonzero entries. For all columns of $\mathbf{X}$ we can extend this as minimization of

$$E = \|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2, \text{ s.t. } L_0(\mathbf{h}_n) \leq L,\ n = 1, \dots, N. \quad (4)$$

Finding the optimal solution for the sparse coding problem is NP-hard [12], where the challenge is to find the optimal atom-to-data assignment, i.e. the locations of the non-zero entries in $\mathbf{H}$. Having this information, the corresponding coefficients *values* are given by the least squares approximation of the columns of $\mathbf{X}$ using the respective assigned atoms.

Many approximate sparse coding approaches have been proposed (see e.g. [13, 14, 15]), where one of the most widely known algorithms is orthogonal matching pursuit (OMP) [16]. OMP is described in Algorithm 1 for a single vector $\mathbf{x}$. In order to find an approximate solution for Eq. (4), we have to repeat this algorithm for each column in $\mathbf{X}$. First, we assign the data vector $\mathbf{x}$ to the residual $\mathbf{r}$. In steps 4-6 we select the atom which approximates the residual $\mathbf{r}$ best, where without loss of generality we assume that the atoms are normalized to unit length. In steps 7-8, the least squares approximation $\hat{\mathbf{x}}$ of the data $\mathbf{x}$, using the atoms selected so far is determined, where $\mathbf{W}_\mathbf{z}$ is the sub dictionary with the atoms depicted by $\mathbf{z}$ and $^+$ denotes the Moore-Penrose inverse. The new residual is defined as $\mathbf{x} - \hat{\mathbf{x}}$. These steps are repeated for L iterations, yielding an index vector $\mathbf{z}$ and coefficients $\mathbf{c}$. The corresponding column in $\mathbf{H}$ is build by setting the entries depicted by $\mathbf{z}$ to the values stored in $\mathbf{c}$, and zeros elsewhere.

Algorithm 1 Orthogonal Matching Pursuit (OMP)

```
1:  $\mathbf{r} \leftarrow \mathbf{x}$
2:  $\mathbf{z} = [\,]$
3:  for $l = 1 : L$ do
4:      $\mathbf{a} = \mathbf{W}^T\,\mathbf{r}$
5:      $z^* = \arg\max |\mathbf{a}|$
6:      $\mathbf{z} \leftarrow [\mathbf{z}, z^*]$
7:      $\mathbf{c} = \mathbf{W}_\mathbf{z}^+\,\mathbf{x}$
8:      $\hat{\mathbf{x}} = \mathbf{W}_\mathbf{z}\,\mathbf{c}$
9:      $\mathbf{r} \leftarrow \mathbf{x} - \hat{\mathbf{x}}$
10: end for
```

2.4. K-SVD

K-SVD [11] is an iterative two stage algorithm which adapts the dictionary $\mathbf{W}$ to a given data set $\mathbf{X}$. In the first stage, the data $\mathbf{X}$ is sparsely coded with fixed dictionary $\mathbf{W}$, i.e. Eq. (4) is minimized with respect to $\mathbf{H}$. This task can be achieved by OMP or any other sparse coding algorithm.

In the second stage the dictionary $\mathbf{W}$ is updated, while holding the atom to data assignment fixed, i.e. the locations of the "nonzeros" in $\mathbf{H}$. The atoms are updated in a random sequence, where $\mathbf{w}_k$ denotes the atom to be updated. The objective can be reformulated as

$$E = \|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2 = \|\mathbf{E}(k) - \mathbf{w}_k\,\mathbf{h}^k\|_F^2, \qquad (5)$$

where $\mathbf{h}^k$ is the k^{th} row of $\mathbf{H}$ and $\mathbf{E}(k) = \mathbf{X} - \sum_{j \neq k} \mathbf{w}_j\,\mathbf{h}^j$. The task is to find vectors $\mathbf{w}_k$ and $\mathbf{h}^k$ in order to minimize

Eq. (5). However, since we do not want to change the locations of the "nonzeros" in $\mathbf{h}^k$, we restrict the problem to those entries (columns) where $\mathbf{h}^k$ is nonzero:

$$\tilde{E} = \|\mathbf{E}(k)_\omega - \mathbf{w}_k\, \mathbf{h}^k_\omega\|_F^2,\ \omega = \{i | h^k_i \neq 0\}. \tag{6}$$

When $\mathbf{u}_1$ and $\mathbf{v}_1$ are the first singular left and right vectors of $\mathbf{E}(k)_\omega$, respectively, and σ_1 is the corresponding first singular value, Eq. (6) becomes minimal when we replace $\mathbf{w}_k$ with $\mathbf{u}_1$ and $\mathbf{h}^k_\omega$ with $\mathbf{v}_1\,\sigma_1$. Since the K atoms are updated using a singular value decomposition (SVD), the algorithm is called K-SVD. This algorithm can be seen as a generalization of k-means, since in k-means each data vector is represented by a single cluster center (atom), while in K-SVD the data is represented as a linear combination of up to L atoms.

2.5. Nonnegative K-SVD (NN K-SVD)

K-SVD can be seen as matrix factorization technique with ℓ^0-sparseness constraints on the columns of $\mathbf{H}$, i.e. $\mathbf{X} \approx \mathbf{W}\,\mathbf{H}, L_0(\mathbf{h}_n) \leq L, n = 1, \ldots, N$. A nonnegative version of K-SVD can therefore be seen as an NMF algorithm with ℓ^0-sparseness constraints on $\mathbf{H}$. Aharon et al. [3] introduced nonnegativity constraints in K-SVD, which is achieved by introducing nonnegativity in the sparse coding stage and the dictionary update stage, respectively. For the sparse coding stage they proposed a nonnegative variant of basis pursuit [15], which replaces the ℓ^0-norm with the ℓ^1-norm. Therefore they used several iterations of the sparse coding algorithm proposed by Hoyer [6]. To obtain an ℓ^0-sparse result, they select the L atoms with largest coefficients from each column of $\mathbf{H}$. Using these atoms, the respective columns of $\mathbf{X}$ are approximated using the nonnegative least squares solver described in [17]. The obtained least squares coefficients replace the original coefficients of the selected atoms in $\mathbf{H}$, while all other coefficients are set to zero. We refer to this algorithm as nonnegative basis pursuit (NN-BP).

The dictionary update resembles standard K-SVD. However, in order to minimize Eq. (6), the SVD is replaced with an iterative SVD approximation, where negative values in $\mathbf{w}_k$ and $\mathbf{h}^k_\omega$ are set to zero after each iteration.

3. NMF WITH ℓ^0-CONSTRAINTS

In this section, we combine the central ideas of NMF, K-SVD and ℓ^1-sparse NMF to obtain two novel algorithms which we call nonnegative matrix factorization with ℓ^0-constraints (NMFℓ^0). Again, for a given nonnegative matrix $\mathbf{X}$ we aim to find nonnegative matrices $\mathbf{W}$ and $\mathbf{H}$, such that $\mathbf{X} \approx \mathbf{W}\,\mathbf{H}$, where $L_0(\mathbf{w}_k) \leq L, k = 1, \ldots, K$, or, $L_0(\mathbf{h}_n) \leq L, n = 1, \ldots, N$. To the algorithm which constrains the bases matrix $\mathbf{W}$, we refer as NMFℓ^0-W, while NMFℓ^0-H denotes the algorithm which constrains $\mathbf{H}$.

Our key observation is that the standard NMF update rules (see Eq. (1-2) and the rules for the Kullback-Leibler divergence in [1]) are *sparseness maintaining*. An entry in $\mathbf{W}$ or $\mathbf{H}$ which is zero before an NMF update, is also zero afterwards, since the update rules consist of *element-wise* products. This means that NMF can always be used to further enhance a sparse solution.

3.1. Sparseness constraints on H (NMFℓ^0-H)

Similar as in the K-SVD framework our algorithm alternates between a sparse coding stage and a dictionary update stage. For sparse coding we can use any arbitrary nonnegative sparse coder, e.g. NN-BP [3]. Alternatively, we can use a modified version of OMP with nonnegativity constraints. We call this algorithm nonnegative matching pursuit (NMP), which is shown in Algorithm 2. When we reinspect Algorithm 1, we see that OMP can violate nonnegativity at two points, namely in step 5, where we select the atom which approximates the residual best, and in step 7, where the data vector is projected into the space spanned by the atoms selected so far. Therefore we introduce $z^* = \arg\max \mathbf{a}$ in step 6 of Algorithm 2, i.e. we drop the absolute value of $\mathbf{a}$. Thus we select an atom whose scalar projection is most probably positive. However, for the case that all entries in $\mathbf{a}$ are negative, the algorithm has to terminate. Secondly, OMP generally violates nonnegativity in the least squares approximation step: $\mathbf{c} = \mathbf{W}^+_\mathbf{z}\,\mathbf{x}$. The multiplication with the Moore-Penrose inverse $\mathbf{W}^+_\mathbf{z}$ usually yields positive *and* negative coefficients. Therefore, we have to replace this step with a nonnegative least squares solution, such as the already mentioned algorithm proposed in [17]. However, for the sake of computational efficiency we use several iterations of the NMF update rule for $\mathbf{H}$ (Eq. (1)) in steps 13-15.

Algorithm 2 Nonnegative Matching Pursuit (NMP)

```
1:  $\mathbf{z} = [\,]$
2:  $\mathbf{c} = [\,]$
3:  $\mathbf{r} \leftarrow \mathbf{x}$
4:  for $l = 1 : L$ do
5:      $\mathbf{a} = \mathbf{W^T r}$
6:      $z^* = \arg\max \mathbf{a}$
7:      $c^* = \max \mathbf{a}$
8:      if $c^* \leq 0$ then
9:          Terminate
10:     end if
11:     $\mathbf{z} \leftarrow [\mathbf{z}, z^*]$
12:     $\mathbf{c} \leftarrow [\mathbf{c}, c^*]$
13:     for $j = 1 : J$ do
14:         $\mathbf{c} \leftarrow \mathbf{c} \otimes \frac{(\mathbf{W}_\mathbf{z}^T \mathbf{x})}{(\mathbf{W}_\mathbf{z}^T \mathbf{W}_\mathbf{z}\,\mathbf{c})}$
15:     end for
16:     $\mathbf{r} \leftarrow \mathbf{x} - \mathbf{W_z c}$
17: end for
```

Once a sparse matrix $\mathbf{H}$ is obtained by NMP, we can simply perform several iterations of the standard NMF update rules for the Euclidean distance measure (Eq. (1-2)) in order to update $\mathbf{W}$ and $\mathbf{H}$. This fulfills exactly our requirements: (i) the objective $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2$ is reduced, (ii) nonnegativity is maintained, (iii) sparseness of $\mathbf{H}$ is maintained. Note that we also update $\mathbf{H}$ in this step, since we want to adapt the coefficient values (i.e. the atom weights) simultaneously with $\mathbf{W}$. However, the atom to data assignment is maintained, i.e. the locations of the "nonzeros". Further, we normalize the atoms to unit length after each update of $\mathbf{W}$. Since it can happen that an atom is $\mathbf{0}$ after an update (e.g. when the atom is not assigned to any data), we reinitialize such an atom uniformly with the value $\sqrt{D}$. NMFℓ^0-H is summarized in Algorithm 3, where I and J denote the number of overall iterations and NMF updates, respectively.

Algorithm 3 NMFℓ^0-H

```
1: Initialize $\mathbf{W}$ randomly
2: for $i = 1 : I$ do
3:     $\mathbf{H} \leftarrow$ sparsely code $\mathbf{X}$ with $\mathbf{W}$ using NMP
4:     for $j = 1 : J$ do
5:         $\mathbf{W} \leftarrow \mathbf{W} \otimes \frac{(\mathbf{X}\,\mathbf{H}^T)}{(\mathbf{W}\,\mathbf{H}\,\mathbf{H}^T)}$
6:         $\mathbf{w}_k \leftarrow \frac{\mathbf{w}_k}{\|\mathbf{w}_k\|}$, $k = 1, \dots, K$
7:         $\mathbf{H} \leftarrow \mathbf{H} \otimes \frac{(\mathbf{W}^T\mathbf{X})}{(\mathbf{W}^T\mathbf{W}\,\mathbf{H})}$
8:     end for
9: end for
```

3.2. Sparseness constraints on W (NMFℓ^0-W)

In order to introduce sparseness constraints in $\mathbf{W}$, we could switch the roles of $\mathbf{W}$ and $\mathbf{H}$ and execute Algorithm 3 with transposed data matrix $\mathbf{X}'$. This would successfully introduce sparseness in the rows of $\mathbf{W}$, in the sense that $L_0(\mathbf{w}^d) \leq L$, $d = 1, \dots, D$, where $\mathbf{w}^d$ is the d^{th} row vector of $\mathbf{W}$. At the same time also the columns of $\mathbf{W}$ would become sparser, since the average number of nonzero entries per column would be maximal $\frac{L\,D}{K}$. Although this technique might be useful too, we want to constrain the *columns* of $\mathbf{W}$, similar as in [2]. This algorithm is presented in Algorithm 4, where again I denotes the number of overall iterations and J is the number of NMF updates. Since the objective $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2$ is convex in $\mathbf{W}$, steps 3-6 yield a close to optimal (unconstrained) $\mathbf{W}$ for given $\mathbf{H}$ and $\mathbf{X}$. Step 7 projects the columns of $\mathbf{W}$ onto the closest vectors whose ℓ^0-norm is less than or equal to L. In steps 8-11 we perform standard NMF, in order to enhance the bases vectors $\mathbf{W}$ and to adapt the coefficient matrix $\mathbf{H}$, maintaining the sparseness of $\mathbf{W}$. In each iteration, the coefficient matrix $\mathbf{H}$ is enhanced and new ℓ^0-sparse bases $\mathbf{W}$ are found. Altogether, the bases of $\mathbf{W}$ are guaranteed to be ℓ^0-sparse and the algorithm converges to a local minimum of $\|\mathbf{X} - \mathbf{W}\,\mathbf{H}\|_F^2$, given that J is sufficiently large.

Algorithm 4 NMFℓ^0-W

```
1: Initialize $\mathbf{H}$ randomly
2: for $i = 1 : I$ do
3:     Set all entries in $\mathbf{W}$ to 1
4:     for $j = 1 : J$ do
5:         $\mathbf{W} \leftarrow \mathbf{W} \otimes \frac{(\mathbf{X}\,\mathbf{H}^T)}{(\mathbf{W}\,\mathbf{H}\,\mathbf{H}^T)}$
6:     end for
7:     Set $D - L$ smallest values in $\mathbf{w}_k$ to zero, $k = 1 \dots K$
8:     for $j = 1 : J$ do
9:         $\mathbf{W} \leftarrow \mathbf{W} \otimes \frac{(\mathbf{X}\,\mathbf{H}^T)}{(\mathbf{W}\,\mathbf{H}\,\mathbf{H}^T)}$
10:        $\mathbf{H} \leftarrow \mathbf{H} \otimes \frac{(\mathbf{W}^T\mathbf{X})}{(\mathbf{W}^T\mathbf{W}\,\mathbf{H})}$
11:    end for
12: end for
```

4. EXPERIMENTS

4.1. Nonnegative Sparse Coding

For our experiments, we generated sparse synthetic data as follows: We created a random dictionary $\mathbf{W}_{\text{true}}$ with $K = 300$ atoms of dimensionality $D = 250$. Each atom was generated by adding 5-10 randomly spread impulses to unit variance Gaussian noise. We filtered the atoms with a low pass, with a cutoff frequency equal to $\frac{1}{8}$ of the sampling frequency. Finally, we discarded the sign and normalized each atom to unit length. Further, we generated a random coefficient matrix $\mathbf{H}_{\text{true}}$ with $L = 10$ nonzero coefficients at random positions in each column. The values of the coefficients were the absolute value of Gaussian noise with variance 10. The sparse synthetic data is given as $\mathbf{X} = \mathbf{W}_{\text{true}}\,\mathbf{H}_{\text{true}}$.

In this way we generated 100 random data sets and executed NN-BP and NMP on each of them, where $\mathbf{W}_{\text{true}}$ was provided to both methods and the allowed number of atoms L was varied between 5 and 15. For NMP we used $J = 30$ NMF iterations to obtain the least-squares coefficients $\mathbf{c}$. We executed NN-BP using 25 (NN-BP$_{25}$) and 1000 (NN-BP$_{1000}$) iterations of the nonnegative sparse coding algorithm [3, 6]. When 1000 iterations are used, the algorithm converges, but it is approximately 40 times slower than NMP (for $L = 15$). When 25 iterations are used, the execution time of NN-BP is roughly the same as for NMP (for the specific values of D, N and K). In Fig. 1 we see the root mean squared error (RMSE) (top) and the number of correctly identified atoms (bottom) as a function of L, averaged over the 100 data sets. The RMSE was calculated according to $\text{RMSE} = \sqrt{\frac{\|\mathbf{X}-\mathbf{W}\,\mathbf{H}\|_F^2}{D\,N}}$. We see that NN-BP$_{1000}$ achieves a lower reconstruction error than NMP and that for $L < 11$ more atoms are identified correctly. Interestingly, NMP identifies more "true" atoms for $L \geq 11$, while the error is still larger than for NN-BP$_{1000}$. It seems that NN-BP manages to identify "more relevant" atoms, i.e. atoms with larger coefficients. On the other hand, we see that the performance of NN-BP$_{25}$ is suboptimal.

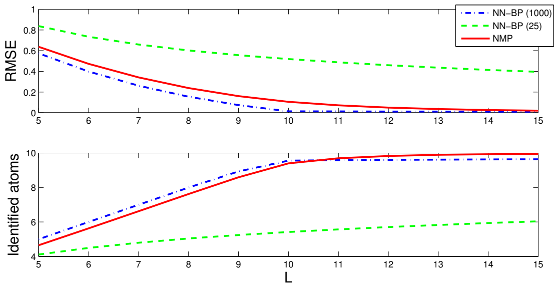

Fig. 1. Performance of sparse coders on synthetic data. Top: Average achieved RMSE. Bottom: Average number of correctly identified atoms. The standard deviation is negligible small.

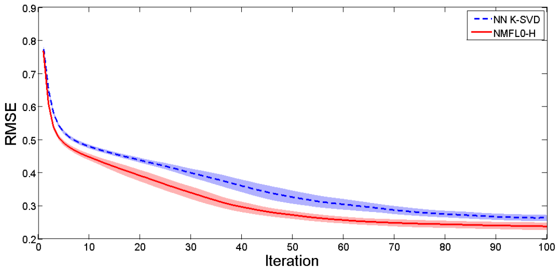

Fig. 2. Average performance of NN K-SVD and NMFℓ^0-H on synthetic data. Shaded bars correspond to standard deviation.

Therefore we can state that NMP exhibits a good trade-off between performance and time consumption.

4.2. NMF with sparseness constraints on H

We applied nonnegative K-SVD and NMFℓ^0-H to the synthetic data sets described in Section 4.1. Both algorithms were provided with the true dictionary size $K = 300$ and sparseness factor $L = 10$. Further, both methods used NMP in the sparse coding step and for both 30 dictionary update steps were performed, i.e. iterative SVD approximations and NMF updates, respectively. Fig. 2 shows the achieved error (averaged over the 100 data sets) as a function of iterations. We see that NMFℓ^0-H converges faster than nonnegative K-SVD and that a slightly better minimum is reached. Next, we conducted an experiment on real-world data, namely on the magnitude spectrogram of 2 minutes of speech from the data base by Cooke et al. [18]. The data matrix $\mathbf{X}$ had the dimensions 513×5656 and we executed nonnegative K-SVD and NMFℓ^0-H for 25 iterations. Both algorithms were started with all parameter combinations out of $K = (100, 200, 300, 400, 500)$, $L = (5, 10, 15, 20)$. Since the different parameter settings yield a strongly varying performance for both algorithms, we determined the relative RMSE according to $\mathrm{RMSE}_{\mathrm{rel}}(K, L, i) = \frac{\mathrm{RMSE}_{\mathrm{NNK-SVD}}(K,L,i)}{\mathrm{RMSE}_{\mathrm{NMF}\ell^0-\mathrm{H}}(K,L,i)}$, where i denotes the iteration count. In Fig. 3 the relative RMSE is shown, averaged over all parameter settings. We see that NMFℓ^0-H converges faster than nonnegative K-SVD, and that a slightly better minimum is reached ($\approx 1\%$).

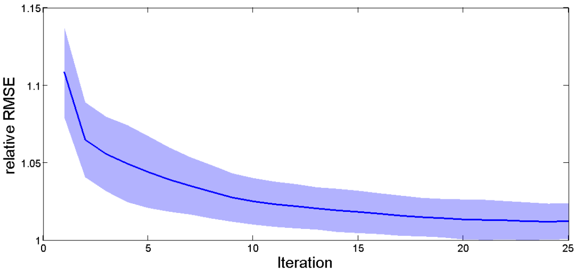

Fig. 3. Average relative RMSE of NN K-SVD compared to NMFℓ^0-H when executed on spectrogram data. Shaded bars correspond to standard deviation.

4.3. NMF with sparseness constraints on W

Hoyer [2] noted, that NMF does not return a part-based representation when applied to the ORL face database [19], since the face images are not aligned. In order to enforce a part-based representation, he constrained the bases vectors to be sparse. We applied NMFℓ^0-W to the ORL database [19], where we used sparseness factors L corresponding to 33%, 25% and 10% of the total pixel number per image (denoted as sparseness classes a, b, c, respectively). We trained 25 bases vectors as in [2], where we executed NMFℓ^0-W for $I = 20$ overall iterations using $J = 30$ NMF updates. Fig. 4 (a, b, c) shows the resulting bases, where dark pixels indicate high values and white pixels indicate low values. In each sparseness class all bases have *exactly* the same number of nonzero pixels (33%, 25% and 10% of total pixels). The average sparseness according to Eq. (3) is 0.54 (a), 0.60 (b) and 0.73 (c). Next, we executed the ℓ^1-sparse NMF algorithm [2] where we constrained the bases to have the same average sparseness as the ℓ^0-sparse bases (0.54, 0.60, 0.73). To achieve satisfying results, at least 2000 iterations were necessary. The resulting bases images are shown in Fig. 4 (d, e, f).

The results for ℓ^1-sparse NMF and NMFℓ^0-W are qualitatively similar, and the representation switches from a global to a local one, when sparseness is increased. We repeated the training 10 times, where neither ℓ^1-sparse NMF nor NMFℓ^0-W seemed to be sensitive to initialization. The average signal-to-reconstruction error ratio is 14.73 dB (a), 14.57 dB (b), 13.89 dB (c) for the ℓ^0-sparse bases, and 15.07 dB (d), 14.95 dB (e), 14.28 dB (f) for the ℓ^1-sparse bases, i.e. ℓ^1-sparse NMF achieves a slightly better reconstruction quality. However, the average percentage of nonzero pixels per ℓ^1-sparse base is 52.35% (d), 43.35% (e) and 19.28% (f), i.e. the ℓ^1-sparse bases contain a significantly larger number of nonzero entries. Further, in a run-time comparison, NMFℓ^0-W was executed about 7 times faster than ℓ^1-sparse NMF.

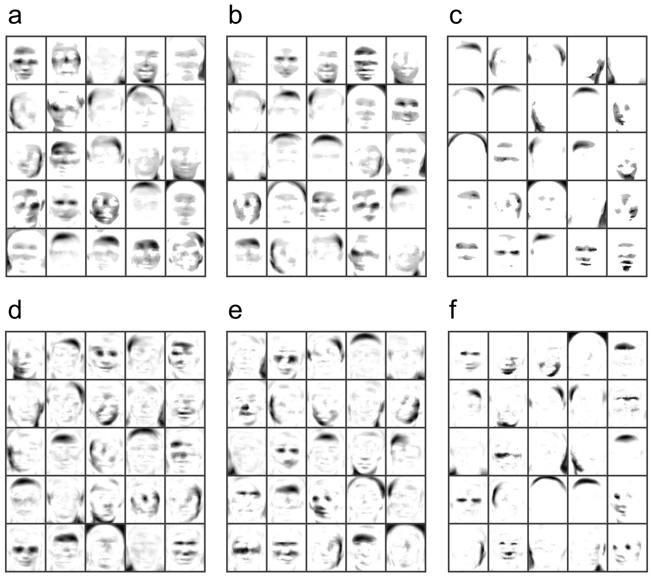

Fig. 4. Top: Bases trained by NMFℓ^0-W. Sparseness factors: (a) 33%, (b) 25%, (c) 10% of total number of pixels per image.
Bottom: Bases trained by ℓ^1-sparse NMF [2]. Sparseness factors according to Eq. (3): (d) 0.54, (e) 0.60, (f) 0.73.

5. CONCLUSION

In this paper we presented two novel algorithms for nonnegative matrix factorization which constrain the columns of the bases matrix and the coefficient matrix, respectively, to have a number of non-zero entries less than or equal to a desired value L. Among other application domains, these techniques are very useful for feature extraction, since the number of features per observation can be limited when we constrain the coefficient matrix. Alternatively, we can also constrain the features to contain nonzero values in no more than L dimensions, which can be interpreted as limited patch size in the case of image data. The key observation for these algorithms is that the classic NMF update rules proposed by Lee and Seung [1] maintain the sparseness of the matrices under optimization. Further, we proposed a nonnegative version of orthogonal matching pursuit, which we call nonnegative matching pursuit (NMP).

In experiments with synthetic sparse data, the nonnegative basis pursuit algorithm proposed in [3] performs slightly better than NMP in terms of reconstruction error. However, NMP offers a good trade-off between execution time and performance, which is essentially important when the sparse coder is required frequently. Experiments on synthetic data and speech spectrograms indicate, that NMFℓ^0-H converges faster than nonnegative K-SVD, and that a slightly better optimum is achieved. Applying NMFℓ^0-W to facial image data shows that a part-based representation is obtained, similar to the results by Hoyer [2]. NMFℓ^0-W achieves almost the same reconstruction quality as ℓ^1-sparse NMF, while using a far smaller number of nonzero entries in the bases.

6. REFERENCES

[1] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556–562, 2001.

[2] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *Journal of Machine Learning Research*, vol. 5, pp. 1457–1469, 2004.

[3] M. Aharon, M. Elad, and A.M. Bruckstein, "K-SVD and its non-negative variant for dictionary design," in *Proceedings of the SPIE conference, Curvelet, Directional, and Sparse Representations II*, 2005, vol. 5914, pp. 11.1–11.13.

[4] D. D. Lee and H. S. Seung, "Learning the parts of objects by nonnegative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[5] K.W. Wilson, B. Raj, P. Smaragdis, and A. Divakaran, "Speech denoising using nonnegative matrix factorization with priors," in *Proceedings of ICASSP*, 2008.

[6] P. O. Hoyer, "Non-negative sparse coding," in *Proceedings of Neural Networks for Signal Processing*, 2002.

[7] J. Eggert and E. Koerner, "Sparse coding and nmf," in *International Joint Conference on Neural Networks*, 2004.

[8] S.Z. Li, X.W. Hou, H.J. Zhang, and Q.S. Cheng, "Learning spatially localized, parts-based representation," in *Proceedings of CVPR*, 2001.

[9] A. Pascual-Montano, J.M. Carazo, K. Kochi, D. Lehmann, and R.D. Pascual-Marqui, "Nonsmooth nonnegative matrix factorization (nsNMF)," in *IEEE Trans. Pattern Analysis Machine Intelligence*, 2006, vol. 87, pp. 1904–1916.

[10] J. A. Tropp, "Just relax: Convex programming methods for identifying sparse signals," *IEEE Trans. Information Theory*, vol. 52, no. 3, pp. 10301051, 2006.

[11] M. Aharon, M. Elad, and A.M. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," in *IEEE Trans. Signal Processing*, 2006, vol. 54, pp. 4311–4322.

[12] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Journal of Constructive Approximation*, vol. 13, pp. 57–98, 1997.

[13] B.D. Rao and K. Kreutz-Delgado, "An affine scaling methodology for best basis selection," in *IEEE Trans. Signal Processing*, 1999, vol. 47, pp. 187–200.

[14] S.G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[15] S.S. Chen, D.L. Donoho, and M.A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1998.

[16] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, 1993.

[17] C.L Lawson and R.J. Hanson, *Solving Least Squares Problems*, Prentice-Hall, 1974.

[18] M. P. Cooke, J. Barker, S. P. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *JASA*, vol. 120, pp. 2421–2424, 2006.

[19] F. S. Samaria and A. Harter, "Parameterisation of a stochastic model for human face identification," in *Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision*, 1994.