Reverse-Convex Programming for Sparse Image Codes

Matthias Heiler and Christoph Schnörr

Computer Vision, Graphics, and Pattern Recognition Group,
Department of Mathematics and Computer Science,
University of Mannheim, 68131 Mannheim, Germany
{heiler, schnoerr}@uni-mannheim.de

Abstract. Reverse-convex programming (RCP) concerns global optimization of a specific class of non-convex optimization problems. We show that a recently proposed model for sparse non-negative matrix factorization (NMF) belongs to this class. Based on this result, we design two algorithms for sparse NMF that solve sequences of convex second-order cone programs (SOCP).

We work out some well-defined modifications of NMF that leave the original model invariant from the optimization viewpoint. They considerably generalize the sparse NMF setting to account for uncertainty in sparseness, for supervised learning, and, by dropping the non-negativity constraint, for sparsity-controlled PCA.

1 Introduction and Related Work

Reverse-convex programming (RCP) is a powerful framework from global optimization which, among others, subsumes d.c. programming [1]. Motivated by a recently proposed model for *sparse non-negative matrix factorization* [2], we employ RCP for solving sparsity-controlled NMF.

NMF was originally proposed to model processes in the physical sciences [3, 4]. In recent years, it has become increasingly popular in machine learning, signal processing, and computer vision as well [5, 6, 7]. One reason for this popularity is that NMF codes naturally favor sparse, parts-based representations [8, 9] which in the context of recognition can be more robust than non-sparse, global features. Especially for computer vision applications, where robustness against occlusion is a constant concern, researchers suggested various extensions of NMF in order to enforce very localized representations [10, 11]. Locality is closely related to *sparseness* which is a desirable property from a machine-learning perspective [12, 13] as well as from biological considerations [14]. We found (Sec. 3) that a particularly accurate sparsity measure introduced in [2] is accurately modeled using *second order conic constraints*.

From a computational viewpoint, second order conic constraints are attractive: Being convex, efficient and robust solvers are available [15, 16] that cover a surprisingly large variety of problems [17]. However, with *sparsity-controlled NMF* difficulties arise since the conic constraints are *reversed*: Admissible are

A. Rangarajan et al. (Eds.): EMMCVPR 2005, LNCS 3757, pp. 600–616, 2005.

points lying *outside* a second order cone. For such *reverse-convex programs* solvers were proposed that find a globally optimal solution [18, 19, 1]. However, to the best of our knowledge, there currently exists no globally optimal algorithm that is practical for solving the large-scale problems common in computer vision and pattern recognition.

In this paper we exploit the geometry of the sparsity-controlled NMF optimization problem to derive two algorithms that efficiently yield *locally* optimal solutions of the respective reverse-convex problems. The algorithms complement existing solvers based on projected gradient descent [2]. However, with our approach, there is *no need to select optimization parameters* (e.g., stepsize) and *performance* is superior in some relevant situations (Sec. 6). In addition, the reverse-convex framework allows to *easily extend the sparsity-controlled NMF model* by additional constraints. As proof-of-concept we show how additional convex constraints can *account for prior knowledge* available in supervised classification. Along a similar line, we present an algorithm for *sparsity-controlled PCA* [20, 21, 22] that uses reverse-convex solvers as subroutine.

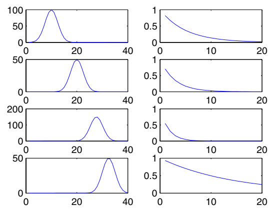
In summary, the contributions of this paper[1] are

– two algorithms based on reverse-convex optimization for solving variations of the NMF problem,
– extensions of the sparsity-controlled NMF model to account for prior knowledge and uncertainty in sparseness, and
– a reverse-convex algorithm for sparsity-controlled PCA.

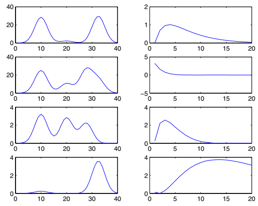
Outline. In Section 2 we present the models considered in this paper. This includes the original model by Hoyer [2] as well as some useful extensions. In Section 3 we explain how, precisely, the sparsity measure used relates to second order cones. Section 4 presents the tangent-plane approach, a fast and practical algorithm based on first-order approximation of the sparsity constraints. Section 5 gives an algorithm based on a dual optimization idea. We present experimental results in Section 6 before we conclude in Section 7.

Notation. For any $m \times n$-matrix M, we denote its i-th column by M_{*i}, and its row by M_{i*}. Unless stated otherwise, $V \in \mathbb{R}_+^{m\times n}$ is a non-negative matrix containing n data points, and $W \in \mathbb{R}_+^{m\times r}$ is a corresponding basis with r-dimensional coefficients $H \in \mathbb{R}_+^{r\times n}$. $\|x\|_p$ denotes the ℓ_p-norm for vectors x, and $\|M\|_F$ the Frobenius norm for matrices: $\|M\|_F = \sqrt{\mathrm{tr}(M^\top M)}$. Further, $\otimes$ denotes Kronecker's matrix product and $\odot$ the element-wise matrix multiplication. $\mathrm{vec}(M)$ is the concatenation of the columns of M. Special matrices we will encounter are $I_{r\times r}$, the $r \times r$ identity matrix, and $E_{a\times b}$, the $a \times b$ matrix with all entries equal to unity. Finally, e is the vector with entries equal to unity.

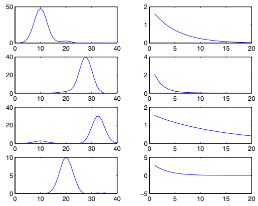
[1] Note that a shorter paper with focus on Computer Vision appears in ICCV. In this shorter paper translation-invariant image bases were treated, but the tangent-plane approach (Sec. 4) and the sparse PCA algorithm (Sec. 5.1) as well as some theoretical and experimental results had to be omitted due to lack of space.



(a) True W and H.



(b) NMF, $s_w^{\min} = 0.0$.



(c) NMF, $s_w^{\min} = 0.6$.

Fig. 1. Paatero experiments. The data set is displayed in Fig. 1(a): Gaussian and exponential distributions are multiplied to yield matrix V. In the experiments, a small amount of Gaussian noise $\eta \sim \mathcal{N}(0, 0.1)$ is added to the product. The results for different values of the min-sparsity constraint are shown in Fig. 1(b) and 1(c): Only a non-trivial sparsity constraint makes recovery of W and H successful.

2 Sparsity Control for NMF and PCA

2.1 Sparse NMF

We consider the NMF optimization problem:

$$\begin{aligned} \min_{W,H} \quad & \|V - WH\|_F^2 \\ \text{s.t.} \quad & 0 \leq W, H. \end{aligned} \tag{1}$$

Although NMF codes are often sparse, it has been suggested to control sparsity by more direct means. This can lead to considerably improved basis functions (Fig. 1). We will employ the following sparseness measure recently introduced in the NMF context by Hoyer [2]:

$$\operatorname{sp}(x) := \frac{1}{\sqrt{n} - 1} \left(\sqrt{n} - \frac{\|x\|_1}{\|x\|_2} \right), \quad x \in \mathbb{R}^n \setminus \{0\}. \tag{2}$$

Since $\frac{1}{\sqrt{n}}\|x\|_1 \leq \|x\|_2 \leq \|x\|_1$, (2) is bounded: $0 \leq \operatorname{sp}(x) \leq 1$. In particular, $\operatorname{sp}(x) = 0$ for minimal sparse vectors with equal non-zero components, and $\operatorname{sp}(x) = 1$ for maximally sparse vectors with all but one vanishing components. By a slight abuse of notation, we will sometimes write $\operatorname{sp}(M) \in \mathbb{R}^n$, meaning $\operatorname{sp}(\cdot)$ is applied to each column of matrix $M \in \mathbb{R}^{m \times n}$.

Originally, it was proposed to use $\operatorname{sp}(x) = \mathit{const.}$ to constrain the set of admissible solutions of (1) [2]. Slightly generalizing, we formulate:

$$\begin{aligned} \min_{W,H} \quad & \|V - WH\|_F^2 \\ \text{s.t.} \quad & 0 \leq W, H \\ & s_w^{\min} \leq \operatorname{sp}(W) \leq s_w^{\max} \\ & s_h^{\min} \leq \operatorname{sp}(H^\top) \leq s_h^{\max}, \end{aligned} \tag{3}$$

where $s_w^{\min}, s_h^{\min}, s_w^{\max}, s_h^{\max}$ are user parameters to control sparsity. To obtain a feasible set with non-void interior, we usually choose $s_{h/w}^{\min}$ strictly smaller than $s_{h/w}^{\max}$.

Alternatively, it can be convenient to trade sparsity for reconstruction accuracy by relaxing the hard constraints:

$$\begin{aligned} \min_{W,H} \quad & \|V - WH\|_F^2 - \lambda_h e^\top \mathrm{sp}(H^\top) - \lambda_w e^\top \mathrm{sp}(W) \\ \text{s.t.} \quad & 0 \le W, H. \end{aligned} \tag{4}$$

Furthermore, for object recognition it is generally useful to integrate available information about object labels into the process of learning basis functions [11]. With our approach it is particularly efficient to restrict, for each class i and for each of its vectors j, the coefficients H_{j*} to a cone around the class center μ_i

$$\begin{aligned} \min_{W,H} \quad & \|V - WH\|_F^2 \\ \text{s.t.} \quad & 0 \le W, H & \text{(5a)} \\ & \|\mu_i - H_{j*}\|_2 \le \lambda \|\mu_i\|_1 \quad \forall i, \forall j \in \mathrm{class}(i). & \text{(5b)} \end{aligned}$$

Given class label information, the μ_i are completely determined by the coefficient matrix H. Thus, they are computed implicitly during optimization.

2.2 Sparse PCA

For data that is non-negative by nature, e.g., image data, certain physical properties, probabilities, or equities, NMF is particularly well-suited. However, in situations where negative values occur we want to allow for negative bases and coefficient vectors as well. This leads to a sparsity-controlled setting similar to PCA [20, 21, 22]. In particular, the problem considered reads

$$\begin{aligned} \min_{W,H} \quad & \|V - WH\|_F^2 \\ \text{s.t.} \quad & s_w^{\min} \le \mathrm{sp}(W) \le s_w^{\max} \\ & s_h^{\min} \le \mathrm{sp}(H^\top) \le s_h^{\max}, \end{aligned} \tag{6}$$

which equals (3) except for the non-negativity constraints that are omitted.

3 Sparsity and Second Order Cones

In this section we show how our sparsity measure relates to second order cones. Through their close link algorithms based on convex programming become useful for sparsity-controlled NMF.

The *second order cone* $\mathcal{L}^{n+1} \subset \mathbb{R}^{n+1}$ is the convex set [17]:

$$\mathcal{L}^{n+1} := \left\{ \begin{pmatrix} x \\ t \end{pmatrix} = (x_1, \ldots, x_n, t)^\top \;\middle|\; \|x\|_2 \le t \right\} , \tag{7}$$

The problem of minimizing a linear objective function, subject to the constraints that several affine functions of the variables are situated in $\mathcal{L}^{n+1}$, is called a *second order cone program (SOCP)*:

$$\begin{aligned}&\inf_{x\in\mathbb{R}^n} f^\top x \\ &\text{s.t.} \begin{pmatrix} A_i x + b_i \\ c_i^\top x + d_i \end{pmatrix} \in \mathcal{L}^{n+1}\,, \qquad i = 1,\dots,m\end{aligned} \tag{8}$$

Note, that linear constraints and, in particular, the condition $x \in \mathbb{R}^n_+$ are important special cases. Our approach to sparsity-constrained NMF, to be developed subsequently, is based on this class of convex optimization problems for which efficient and robust solvers exist in software [15, 16].

On the non-negative cone we can model the sparseness-measure (2) using the family of *convex* sets parametrized by sparsity-parameter $s \in [0, 1]$:

$$C(s) := \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{pmatrix} x \\ e^\top x / c_{n,s} \end{pmatrix} \in \mathcal{L}^{n+1} \right\}\,, \quad c_{n,s} := \sqrt{n} - (\sqrt{n} - 1)s. \tag{9}$$

Inserting the bounds $0 \le \mathrm{sp}(x) \le 1$ for s, we obtain

$$C(0) = \big\{ \lambda e\,,\ 0 < \lambda \in \mathbb{R} \big\} \qquad \text{and} \qquad \mathbb{R}^n_+ \subset C(1). \tag{10}$$

This raises the question as to when we must impose non-negativity constraints explicitly.

Proposition 1. *The set $C(s)$ contains non-positive vectors $x \neq 0$ if:*

$$\frac{\sqrt{n} - \sqrt{n-1}}{\sqrt{n} - 1} < s \le 1\,, \quad n \ge 3 \tag{11}$$

Proof. We observe that if $x \in C(s)$, then $\lambda x \in C(s)$ for arbitrary $0 < \lambda \in \mathbb{R}$, because $\|\lambda x\|_2 - e^\top(\lambda x)/c_{n,s} = \lambda(\|x\|_2 - e^\top x/c_{n,s}) \le 0$. Hence it suffices to consider vectors x with $\|x\|_2 = 1$. According to definition (9), such vectors tend to be in $C(s)$ the more they are aligned with e. Therefore, w.l.o.g., put $x_n = 0$ and $x_i = (n-1)^{-1/2}\,,\ i = 1,\dots,n-1$. Then $x \in C(s)$ if $c_{n,s} < \sqrt{n-1}$, and the result follows from the definition of $c_{n,s}$ in (9). Finally, for $n = 2$ the lower bound for s equals 1, i.e., no non-positive vectors exist for all admissible values of s. □

This argument shows that $C(s') \subseteq C(s)$ for $s' \le s$. To represent the feasible set of problem (3), we combine the convex non-negativity condition with the convex upper bound constraint:

$$\big\{ x \in \mathbb{R}^n_+ \;\big|\; \mathrm{sp}(x) \le s \big\} \;=\; \mathbb{R}^n_+ \cap C(s)\,, \tag{12}$$

and impose the *non-convex* lower bound constraint by subsequently removing $C(s')$:

$$\big\{ x \in \mathbb{R}^n_+ \;\big|\; s' \le \mathrm{sp}(x) \le s\,,\ s' < s \big\} \;=\; \big(\mathbb{R}^n_+ \cap C(s)\big) \setminus C(s') \tag{13}$$

To reformulate (3), we define accordingly, based on (9):

$$\mathcal{C}_w(s) := \{W \in \mathbb{R}^{m\times r} \mid W_{*i} \in C(s)\,,\ i = 1,\dots,r\} \tag{14}$$
$$\mathcal{C}_h(s) := \{H \in \mathbb{R}^{r\times n} \mid H_{i*} \in C(s)\,,\ i = 1,\dots,r\} \tag{15}$$

As a result, the sparsity-constrained NMF problem (3) now reads:

$$\begin{aligned}\min_{W,H}\quad & \|V - WH\|_F^2 \\ \text{s.t.}\quad & W \in \big(\mathbb{R}_+^{m\times r} \cap \mathcal{C}_w(s_w^{\max})\big) \setminus \mathcal{C}_w(s_w^{\min}) \\ & H \in \big(\mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max})\big) \setminus \mathcal{C}_h(s_h^{\min})\end{aligned} \tag{16}$$

This formulation makes explicit that enforcing sparse NMF solutions introduces a single additional *reverse-convex* constraint for W and H, respectively. Consequently, not only the joint optimization of (W, H) is non-convex, but individual optimization of W and H are also.

4 Tangent-Plane Approach

In this section, we present an optimization scheme for sparsity-controlled NMF which relies on linear approximations of the reverse-convex constraint in (16). We process W and H alternately, leaving one fixed while optimizing the other. For presentation, we consider the H-step:

$$\begin{aligned}\min_{H}\quad & f(H) = \|V - WH\|_F^2 \\ \text{s.t.}\quad & H \in \big(\mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max})\big) \setminus \mathcal{C}_h(s_h^{\min}).\end{aligned} \tag{17}$$

The W-step is analogous. Throughout this paper, we assume $s_h^{\min} < s_h^{\max}$, i.e., the interior of the feasible set is non-empty. Furthermore, we assume that every basis vector W_{*i} contains at least one non-zero entry.

Step one. The algorithm starts by setting $s_h^{\min} = 0$ in (17), and by computing the global optimum of the convex problem: $\min f(H), H \in \mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max})$, denoted by $\tilde{H}^0$. Rewriting the objective function:

$$\begin{aligned} f(H) &= \|V^\top - H^\top W^\top\|_F^2 \\ &= \|\mathrm{vec}(V^\top) - (W \otimes I)\mathrm{vec}(H^\top)\|_2^2\,,\end{aligned} \tag{18}$$

we observe that $\tilde{H}^0$ solves the SOCP:

$$\min_{\mathbb{R}}\ t\,,\quad H \in \mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max})\,,\quad \begin{pmatrix}\mathrm{vec}(V^\top) - (W \otimes I)\mathrm{vec}(H^\top) \\ t\end{pmatrix} \in \mathcal{L}^{r\cdot n+1}. \tag{19}$$

Note that $\tilde{H}^0$ will in general be infeasible w.r.t. the original problem because the reverse-convex constraint of (17) is not imposed in (19). In case $\tilde{H}^0$ already is feasible the reverse-convex min-sparsity constraint is superfluous and the algorithm terminates returning $\tilde{H}^0$ as solution.

Table 1. Tangent-plane approximation algorithm in pseudocode

1.)	$H^0 \leftarrow$ solution of (19), $J^0 \leftarrow \emptyset,\ k \leftarrow 0$
2.)	repeat
3.)	$\quad \tilde{H}^k \leftarrow H^k$
4.)	$\quad$ repeat
5.)	$\qquad J^k \leftarrow J^k \cup \{j \in 1,\dots,r : \tilde{H}^k_{j*} \in \mathcal{C}_h(s_h^{\min})\}$
6.)	$\qquad t_j^k \leftarrow \nabla\mathcal{C}(s_h^{\min})(\pi(\tilde{H}^k_{j*})) \quad \forall j \in J^k$
7.)	$\qquad \tilde{H}^k \leftarrow$ solution of (20) replacing H^k by $\tilde{H}^k$
8.)	$\quad$ until $\tilde{H}^k_{j*}$ feasible
9.)	$\quad H^{k+1} \leftarrow \tilde{H}^k,\ J^{k+1} \leftarrow J^k,\ k \leftarrow k+1$
10.)	until $\lvert f(H^k) - f(H^{k-1})\rvert \leq \epsilon$

Step two. Using an iteration counter k initialized to 0 we determine in step two the index set $J^k \subseteq \{1,\dots,r\}$ of those vectors $\tilde{H}^k_{j*}$ violating the reverse-convex constraint, that is $\tilde{H}^k_{j*} \in C(s_h^{\min})$. Let $\pi(\tilde{H}^k_{j*})$ denote the projections of $\tilde{H}^k_{j*}$ onto $\partial C(s_h^{\min})$, $\forall j \in J^k$, and t_j^k the normals to $\mathcal{C}_h(s_h^{\min})$ at these points, and H^k the matrix $\tilde{H}^k$ rectified accordingly. Given J^k, we re-solve (19) with additional linear constraints enforcing feasibility of each H^k_{j*} , $j \in J^k$:

$$\begin{aligned} &\min_{\mathbb{R}} t\,, \quad H \in \mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max})\,, \quad \begin{pmatrix} \mathrm{vec}(V^\top) - (W \otimes I)\mathrm{vec}(H^\top) \\ t \end{pmatrix} \in \mathcal{L}^{r\cdot n+1} \\ &\qquad \big\langle t_j^k, H_{j*} - \pi(H^k_{j*})\big\rangle \geq 0\,, \quad \forall j \in J^k \end{aligned} \tag{20}$$

Geometrically, the linear constraints force $H^k_{j*}, j \in J^k$ onto the non-negative side of the plane tangential to the min-sparsity cone running through $\pi(H^k_{j*})$. Thus, in the solution $\tilde{H}^{k+1}$ of (20) all $\tilde{H}^{k+1}_{j*}$, $j \in J^k$, will be feasible. However, it is possible that new vectors $\tilde{H}^{k+1}_{j*}$ corresponding to unconstrained indices $j \notin J^k$ have become infeasible. When this happens we augment J^k accordingly, adjust the tangent planes to reflect the new position of $\tilde{H}^{k+1}_{j*}$, and re-solve (20) until the solution is feasible. The result is denoted by H^{k+1}, the corresponding index set by J^{k+1}. Finally, we increment the iteration counter: $k \leftarrow k+1$ and check whether H^{k+1} satisfies the termination criterion $\big|f(H^{k+1}) - f(H^k)\big| \leq \epsilon$. If it does not we continue with step two. The algorithm is summarized in Tab. 1.

4.1 Convergence Properties

In the following discussion we will use matrices $T^k = (t_j^k)_{j\in J}$ that have tangent plane vector t_j^k as j-th column when $j \in J^k$ and zeros elsewhere.

Proposition 2. *If the cone constraints are regular*[2] *the tangent-plane algorithm yields a sequence $H^1, H^2, \dots$ of feasible points, every cluster point of which is a local optimum.*

Proof. (Sketch). Our proof follows [18–Prop. 3.2]. First, note that for every $k > 0$ the solution H^k of iteration k is a feasible point for the SOCP solved in iteration $k+1$. Therefore, $\{f(H^k)\}_{k=1,\dots}$ is a decreasing sequence, bounded from below and thus convergent. By assumption, no column of $W \geq 0$ equals the zero vector. Then, $\{H : f(H) \leq f(H^k)\}$ is bounded for each k. Consequently, the sequence $\{H^k\}_{k=1,\dots}$ of solutions of (20) and the corresponding sequences $\{T^k\}_{k=1,\dots}$ of tangent planes are bounded and contain converging subsequences. Let $\{H^{k_\nu}\}_{\nu=1,\dots}$ and $\{T^{k_\nu}\}_{\nu=1,\dots}$ be subsequences converging to cluster points $\bar{H}$ and $\bar{T}$.

Because H^{k_ν} is the global solution of a convex program we have

$$f(H^{k_\nu}) \leq f(H), \qquad \forall H \in \mathcal{C}(s_h^{\max}) \text{ with } T^{k_\nu \top} H \geq 0, \tag{21}$$

and in the limit $\nu \to \infty$

$$f(\bar{H}) \leq f(H), \qquad \forall H \in \mathcal{C}(s_h^{\max}) \text{ with } \bar{T}^\top H \geq 0. \tag{22}$$

We assumed the tangent plane constraints are regular. Then the constraints active in $\bar{T}$ correspond to entries $\bar{H}_{j*} \in \partial \mathcal{C}_h(s_h^{\min}), j \in J$. According to (22) there is no feasible descent direction at $\bar{H}$ and, thus, it must be a stationary point. Since the target function is quadratic positive-semidefinite it is an optimum. □

While the individual optimization of H for given W converges (and vice versa), the same cannot be claimed for the *alternating* sequence of optimizations in W and H necessary to solve (3): The intervening optimization of W prevents us from deriving a bound on $f(H)$ from a previously found locally optimal H. In rare cases, this can lead to undesirable oscillations.

4.2 Practical Remarks

Two things are remarkable regarding the tangent-plane algorithm: First, multiple tangent-planes with reversed signs can also be used to approximate the convex max-sparsity constraints. Then (20) reduces to a *quadratic programming* (QP) problem. Except for linear programming, QP solvers are often among the most efficient mathematical programming codes available. Thus, a speedup might be gained by using QP instead of SOCP solvers, in particular, for the important case when no max-sparsity constraints are specified (i.e., $s_h^{\max} = s_w^{\max} = 1$).

A second remark concerns the termination criterion (step 10 in Tab. 1). While it can be chosen almost arbitrarily rigid, an overly small ϵ does not help in the overall optimization w.r.t. W *and* H. As long as, e.g., W is known only approximately, there is no need to compute the corresponding H to the last digit. In Sec. 6 we chose relatively large ϵ so that the outer loop (steps 2 to 10 in Tab. 1) executed only once or twice before the variable was switched.

[2] According to Proposition 1 this is the case iff $s \neq \frac{\sqrt{n}-\sqrt{n-1}}{\sqrt{n}-1}$ and the interior of the feasible set is non-empty.

5 Sparsity-Maximization Approach

In this section we present an algorithm that alternately maximizes sparsity and minimizes the objective function, in effect replacing the reverse-convex constraint by a sequence of convex proximity constraints. Unlike the tangent-plane approach presented above, this algorithm can easily be employed to yield monotonically decreasing sequences of objective values in H *and* in W, i.e., $f(H^k, W^k) \geq f(H^{k+1}, W^k) \geq f(H^{k+1}, W^{k+1}) \geq \cdots$, with $f(H, W) = \|V - WH\|_F^2$. This rules out oscillations that are rare, but cumbersome to avoid with the tangent-plane approach.

5.1 Sparse NMF

In order to solve (16) we again alternately optimize for W and for H, keeping the other constant. Since both optimizations are symmetric, we focus our presentation on the H step.

Our algorithm is motivated by results from global optimization [18] and consists of two complementary steps: One maximizes sparsity subject to the constraint that the objective value must not increase. Dually, the other optimizes the objective function $f(H) = \|V - WH\|_F^2$ under the condition that the min-sparsity constraint may not be violated.

Initialization. Let $H^0 \in \partial\mathcal{C}_h(s_h^{\min})$ be an initialization on the boundary of the min-sparsity cone. It may be computed by solving (19) for H, i.e., ignoring min-sparsity constraints and projecting the solution onto $\partial\mathcal{C}_h(s_h^{\min})$. Set $k \leftarrow 0$.

Step one. In the first step, consider the program

$$\begin{aligned} \max_{H} \quad & g(H) = \min_{j} \{\mathrm{sp}(H_{j*})\} \\ \text{s.t.} \quad & H \in \mathbb{R}_+^{r \times n} \cap \mathcal{C}_h(s_h^{\max}) \\ & f(H) \leq f(H^k) \end{aligned} \tag{23}$$

that maximizes sparsity of the least sparse H_{j*} subject to the constraint that the solution may not measure worse than H^k in terms of the target function f. This is a convex maximization problem on a bounded domain. As such, it can in principle be solved to global optimality [18]. However, practical algorithms exist for small-scale problems only.

Thus, we will content ourselves with a local improvement that is obtained by replacing $\mathrm{sp}(x)$ by its first order Taylor expansion at H^k, resulting in the SOCP

$$\begin{aligned} \max_{H,t} \quad & t \\ \text{s.t.} \quad & H \in \mathbb{R}_+^{r \times n} \cap \mathcal{C}_h(s_h^{\max}) \\ & \begin{pmatrix} \mathrm{vec}(V^\top) - (W \otimes I)\mathrm{vec}(H^\top) \\ f(H^k) \end{pmatrix} \in \mathcal{L}^{rn+1} \\ & t \leq \mathrm{sp}(H_{j*}) + \langle \nabla_{H_{j*}} \mathrm{sp}(H_{j*}^k)^\top, H_{j*} - H_{j*}^k \rangle \quad \forall j. \end{aligned} \tag{24}$$

Table 2. Sparsity-maximization algorithm in pseudocode

1.)	$H^0 \leftarrow$ solution of (19) projected on $\partial\mathcal{C}_h(s_h^{\min})$, $k \leftarrow 0$
2.)	repeat
3.)	$\quad H^{\mathrm{sp}} \leftarrow$ solution of (24)
4.)	$\quad H^{k+1} \leftarrow$ solution of (25)
5.)	$\quad k \leftarrow k+1$
6.)	until $\lvert f(H^k) - f(H^{k-1})\rvert \leq \epsilon$

Let H^{sp} denote the corresponding solution. Note that H^k is a feasible point of (24) and the sparsity cone is convex. Thus, optimization will in fact yield $g(H^{\mathrm{sp}}) \geq g(H^k)$.

Step two. In the second step we solve the SOCP

$$\begin{aligned} \min_{H,t} \quad & t \\ \text{s.t.} \quad & \begin{pmatrix} \mathrm{vec}(V^\top) - (W \otimes I)\mathrm{vec}(H^\top) \\ t \end{pmatrix} \in \mathcal{L}^{rn+1} \\ & \begin{pmatrix} H_{j*} - H_{j*}^{\mathrm{sp}} \\ \min_{q \in \mathcal{C}(s_h^{\min})} \|q - H_{j*}^{\mathrm{sp}}\|_2 \end{pmatrix} \in \mathcal{L}^{n+1} \quad \forall j \\ & H \in \mathbb{R}_+^{r\times n} \cap \mathcal{C}_h(s_h^{\max}). \end{aligned} \tag{25}$$

This problem is identical to (16) restricted to H, except for the reverse-convex min-sparsity constraint that is replaced by a convex proximity constraint: Each H_{j*} is restricted to lie within a "save" distance from H_{j*}^{sp}.

Akin to [18], convergence to a local optimum can be proven under mild restrictions (notably, that W is not degenerate and contains at least one non-zero entry in each column). A proof will be presented in an extended paper.

Assuming convergence for a moment, it is easy to see that for any given feasible H^0 the algorithm terminates with an H^k that performs at least as well as H^0 in terms of f, i.e., $f(H^k) \leq f(H^0)$. This is, because in each iteration i the current estimate H^i is a feasible point of the convex program (25) minimizing f so that $f(H^{i+1}) \leq f(H^i)$, $i = 0, \ldots, k-1$. When the sparsity-maximization algorithm runs multiple times, alternately optimizing H for fixed W and vice versa, one will, after the first iteration, not initialize the algorithm by solving (19) and projecting subsequently, but use the current estimates for H and W as initializations. This way, one obtains a monotonically decreasing sequence of objective values $\|V - W^k H^k\|_F^2$.

Relaxed Formulation. The relaxed form of sparsity-controlled NMF described in (4) is optimized similarly by linearizing $\mathrm{sp}(x)$ around H^k, yielding the SOCP

$$\begin{aligned}
\min_{H,t,s} \quad & t - \lambda_h e^\top s \\
\text{s.t.} \quad & \begin{pmatrix} \operatorname{vec}(V^\top) - (W \otimes I)\operatorname{vec}(H^\top) \\ t \end{pmatrix} \in \mathcal{L}^{rn+1} \\
& s_j \leq \operatorname{sp}(H^k_{j*}) + \langle \nabla_{H_{j*}} \operatorname{sp}(H^k_{j*})^\top, H_{j*} - H^k_{j*} \rangle \quad \forall j \\
& H \in \mathbb{R}^{r\times n}_+ .
\end{aligned} \tag{26}$$

Thus, in order to solve (4) for H we iteratively solve instances of (26) until convergence.

Exploiting Information from Class Labels. The supervised variant (5) of the NMF problem is readily solved by above algorithms since (5b) translates, for each class i and for each coefficient vector H_{j*} belonging to class i, into a second order constraint

$$\begin{pmatrix} 1/n_i H_{(i)} e - H_{j*} \\ \lambda/n_i e^\top H_{(i)} e \end{pmatrix} \in \mathcal{L}^{n+1}, \quad \forall i, \forall j \in \operatorname{class}(i). \tag{27}$$

Here, the $r \times n_i$-matrix of coefficients belonging to class i is abbreviated $H_{(i)}$ and we recognize $\mu_i = 1/n_i H_{(i)} e$. Adding these constraints to, e.g., (24) and (25) yields an algorithm for solving supervised NMF.

5.2 Sparse PCA

Next, we show how to optimize for H when both, W and H may contain negative entries. The idea is that we can factorize any non-zero matrix $M \in \mathbb{R}^{m\times n}$ into $M_\pm \equiv \operatorname{sign}(M) \in \mathbb{R}^{m\times n}$ and $M_+ \in \mathbb{R}^{m\times n}_+$ s.t. $M = M_\pm \odot M_+$. Since sparsity is not affected by sign changes or multiplicative constants we observe

$$\operatorname{sp}(M) = \operatorname{sp}(M_+), \tag{28}$$

i.e., it is *sufficient to exercise sparsity control on the non-negative part* of x. Thus, the sparsity-controlled NMF algorithms presented above can be used on W_+ and H_+. Finally, for those entries in W and H that are close to 0 we subsequently optimize signs using convex programming.

Step one. E.g., in the H-step we first optimize for H_+, by solving

$$\begin{aligned}
\min_{H_+,t} \quad & t \\
\text{s.t.} \quad & \begin{pmatrix} \operatorname{vec}(V) - ((I \otimes W) \odot H_S)\operatorname{vec}(H_+) \\ t \end{pmatrix} \in \mathcal{L}^{rn+1} \\
& H_+ \in (\mathbb{R}^{r\times n}_+ \cap \mathcal{C}_h(s_h^{\max})) \setminus \mathcal{C}_h(s_h^{\min})
\end{aligned} \tag{29}$$

using any of the techniques presented above. Note that (29) is identical to the NMF case, i.e., it minimizes the original problem for $H = H_\pm \odot H_+$ but the signs of H are not allowed to change. The matrix H_S is given by

$$H_S = (I_{r\times r} \otimes E_{n\times n}) \odot (\operatorname{vec}(H_\pm) e^\top)^\top . \tag{30}$$

Step two. In a second step, we solve for $H_\pm$ using the convex program

$$\begin{aligned} \min_{t, H_\pm \in H_\epsilon} \quad & t \\ \text{s.t.} \quad & \begin{pmatrix} \mathrm{vec}(V) - ((I \otimes W) \odot H_A)\mathrm{vec}(H_\pm) \\ t \end{pmatrix} \in \mathcal{L}^{rn+1} \\ & -1 \le H_\pm \le 1, \end{aligned} \tag{31}$$

where H_A is constructed from H_+ analogously to (30):

$$H_A = (I_{r\times r} \otimes E_{n\times n}) \odot (\mathrm{vec}(H_+)e^\top)^\top. \tag{32}$$

H_ϵ denotes those entries in H_+ that are within ϵ from 0. Entries in $H_\pm$ corresponding to larger entries in H_+ are *not* optimized in order to prevent an entry in $H_\pm$ with small norm cancel out an entry with large norm in H_+, thus possibly modifying sparseness of the product $H = H_\pm \odot H_+$.

6 Experiments

6.1 Comparison with Established Algorithms

To see how our algorithms compare against an established method we computed sparsity-controlled decompositions into $r = 4$ basis functions for a subset of the USPS handwritten digits data set using our methods and projected gradient descent (pgd) as proposed in [2]. For different choices of sparseness we report mean and standard deviation of the runtime and mean residual error[3] averaged over 10 runs in Tab. 3. Note that the stopping criterion used was different for our algorithms and for pgd: We stopped when after a full iteration the objective value did not improve at least by a constant, the pgd implementation used[4] stopped as soon as the norm of the gradient was smaller than some ϵ. As the error measurements shown in Tab. 3 demonstrate, both stopping criteria yield comparable results. Regarding running time we see that the tangent-plane approach was usually fastest, followed by sparse-maximization. Also, our algorithms usually showed relatively small variation between individual runs while the runtime of pgd varied strongly, dependent on the randomly chosen starting points.

6.2 Global Approaches

A potential source of difficulties with the sparsity-maximization algorithm is that the lower bound on sparsity is optimized only locally in (24). Through the proximity constraint in (25) the amount of sparsity obtained in effect limits the

[3] Standard deviation of the residual error was equally negligible for all algorithms.

[4] We used the pgd code kindly provided by the author of [2], and removed all logging and monitoring parts to speed up calculation. Our SOCP solver was Mosek 3.2 from MOSEK ApS, Denmark, running under Linux.

Table 3. Performance comparison. Comparison of the tangent-plane (tgp) approach and the sparsity-maximization algorithm (spm) with projected gradient descent (pgd). Sparse decompositions of the digit data set were computed. Statistics collected over 10 repeated runs are reported for runtime (sec.) and residual error $\|V - WH\|_F^2$. tgp and spm are usually faster than pgd while keeping errors small.

sparsity	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
mean time tgp	34.35	35.46	41.30	64.97	66.80	60.86	56.33	51.82	42.50
mean time spm	94.81	106.20	133.52	159.30	173.14	167.06	114.36	78.42	74.96
mean time pgd	517.02	1038.99	218.17	70.24	177.35	189.48	167.94	430.36	322.88
stdv time tgp	3.17	2.64	3.84	5.50	8.51	7.05	4.25	0.76	0.38
stdv time spm	3.48	12.67	23.67	16.45	11.51	11.64	7.69	1.22	1.29
stdv time pgd	278.24	21.21	128.00	8.90	78.12	52.54	95.53	439.34	174.81
mean error tgp	0.82	0.76	0.73	0.72	0.78	0.89	0.99	1.08	1.12
mean error spm	0.81	0.77	0.74	0.73	0.78	0.89	1.00	1.08	1.13
mean error pgd	0.85	0.79	0.74	0.72	0.77	0.88	0.99	1.07	1.12

step size of the algorithm. Insufficient sparsity optimization may, in the worst case, lead to convergence to a bad local optimum.

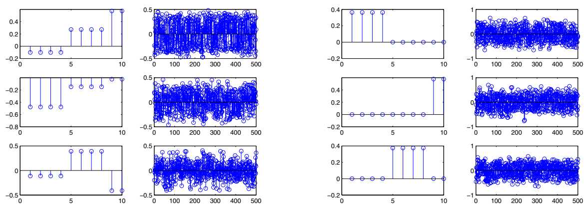
To see if this worst-case scenario is relevant in practice, we discretized the problem by sampling the sparsity cones using rotated and scaled version of the current estimate H^k and then evaluated g in (23) using samples from each individual sparsity cone. Then we picked one sample from each cone and computed (24) replacing the starting point H^k by the sampled coordinates. For an exhaustive search on r cones each sampled with s points we have s^r starting points to consider.

For demonstration we used the artificial Paatero data set [23] consisting of products of Gaussian and exponential functions (Fig. 1). This data set is suitable since it is not overly large and sparsity control is crucial for its successful factorization (cf. [23] and Fig. 1).

In the sparsity-maximization algorithm we first sampled the four sparsity cones corresponding to each basis function of the data for $s_w \geq 0.6$ sparsely, using only 10 rotations on each cone. We then combined the samples on each cone in each possible way and evaluated g for all corresponding starting points. In a second experiment we placed 1000 points on each sparsity cone, and randomly selected 10^4 combinations as starting points. The best results obtained over four runs and 80 iterations with our local linearization method and the sparse enumeration (first) and the sampling (second) strategy, are reported below:

Algorithm	min-sparsity	objective value
local linearization	0.60	0.24
sparse enumeration	0.60	0.26
sampling	0.60	0.26

We see that the local sparsity maximization yields results comparable to the sampling strategies. In fact, it is better: Over four repeated runs with each of the sampling strategies we observed outliers with very bad objective values (not



(a) PCA (b) Sparse PCA

Fig. 2. Sparse PCA experiment. Basis and coefficients for an artificial data set are shown. Only sparsity-controlled PCA successfully recovers the structure of the data.

shown). This is most likely caused by severe under-sampling of the sparsity cones. This problem is not straightforward to circumvent: With above sampling schemes a run over 80 iterations takes about 24h of computing[5], so more sampling is not an option. In comparison, the proposed algorithm finishes in few seconds.

6.3 Sparse PCA

As proof-of-concept we factorized the artificial data set examined in [20] using PCA and sparsity-controlled PCA. The data set consists of three factors sampled from $V_1 \sim \mathcal{N}(0, 290), V_2 \sim \mathcal{N}(0, 300), V_3 \sim -0.3V_1 + 0.925V_2 + \eta$ and additional Gaussian noise. The sparse PCA algorithm iteratively solved (29) and (31) using the relaxed optimization framework (26) with $\lambda_w = 0.6$ and a constraint limiting the admissible reconstruction error.

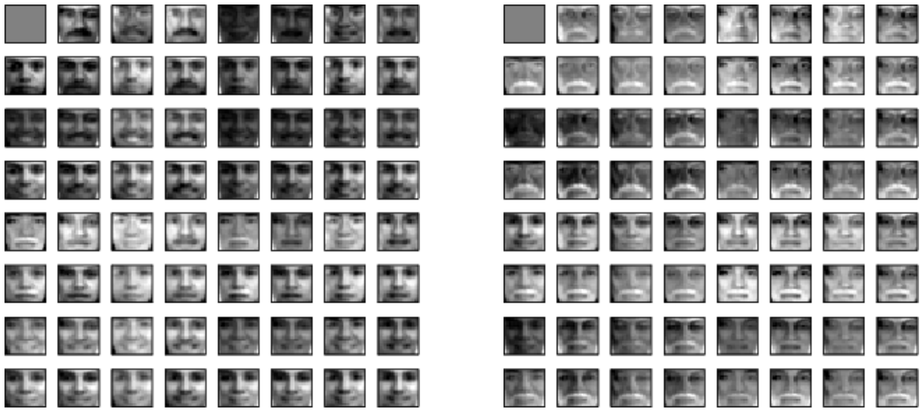
In Fig. 2 we depict the factors and factor-loadings for PCA and sparsity-controlled PCA (best result out of three repeated runs). It is apparent that sparsity-controlled PCA correctly factorizes the data, while classical PCA fails.

6.4 Modeling a Low-Entropy Image Class

A sample application using real-world data is face modeling: Human faces, aligned, cropped and evenly lit, lead to highly structured images. With such images, sparse NMF appears robust against quantization: We learned a sparse image code ($r = 6, s_h^{\min} = 0.3$) for face images [24] and a PCA code for comparison. Then we enumerated possible reconstructions by setting each entry of the coefficient vector to 0 or to 1. The resulting $2^6 = 64$ images are shown in Fig. 3: While most NMF "reconstructions" look remarkably natural the corresponding PCA images mostly suffer severe degradation.

To measure qualitatively how quantization affects reconstruction performance we used PCA and NMF to find a large image base ($r = 100$) on a subset of the face data. Then, we quantized the reconstruction coefficients H using k-means on each individual row of coefficients H_{j*}. The results are shown

[5] On machines with 3GHz P4, 2GB RAM, running Matlab under Linux.



(a) NMF: $W \cdot \delta(h > \epsilon)$ (b) PCA: $W \cdot \delta(|h| > \epsilon)$

Fig. 3. Robustness against quantization. The 64 faces corresponding to previously learned 6bit NMF/PCA image codes after quantization of the coefficients. The top left image corresponds to the binary coefficient vector $\#b000000$, the bottom right image to $\#b111111$. The NMF faces suffer less from quantization than their PCA counterparts.

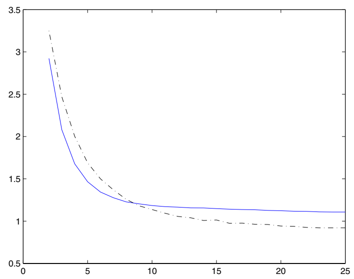


Fig. 4. Robustness against quantization. An basis for face images was trained using NMF (solid blue line) and PCA (dashed black line). The reconstruction coefficients H where quantized using k-means for $k = 2, \dots, 25$ (x-axis) and the reconstruction error $f(W, H)$ was determined (y-axis) for the training data set. NMF is more robust against strong quantization.

in Fig. 4: As expected, PCA offers slightly better reconstruction performance for large values of k. With stronger quantization, however, it looses its advantage and NMF performs better.

This is surprising as quantization robustness was not an original design goal of NMF codes. From a Bayesian perspective we can explain this result by the fact that as quantization (or noise) increases the influence of prior information becomes more important. NMF models the prior information that images are non-negative. PCA has no such constraint and thus suffers more from strong quantization.

6.5 Supervised Training

To show that the supervised label constraints (5b) can be useful we trained NMF codes ($r = 4$) on only 100 samples from the USPS handwritten digit data set. We used different values for the parameter λ and a very simple conditional maximum entropy model $p(y|h)$ with mean coefficient values $\mathbb{E}[h_i]$ as only features for classification. The number of errors on a 300 sample test dataset is given below:

λ	1e4	1e2	1	1e-2	1e-4	1e-6
#errors	108	82	75	60	58	56

When λ is large, i.e., the supervised label constraint is inactive, the error is about 36% (108 out of 300 samples). This is slightly worse than a corresponding PCA basis (95 errors) would achieve. As the label constraint is strengthened the classification performance improves and finally is almost twice as good as in the unsupervised case.

7 Conclusion

We presented two algorithms for sparse coding based on ideas from reverse-convex programming and non-negative matrix factorization. The algorithms are conceptually clean, easy to use (no free parameters), and show attractive performance characteristics. Most importantly, they are flexible enough to be extended along various dimensions. For instance, prior information can be accounted for by adding a single conic constraint, and sparsity-controlled PCA is possible by separate optimization of absolute value and signs of the factors.

References

1. R. Horst and H. Tuy, *Global Optimization.* Springer, Berlin, 1996.
2. P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *J. of Mach. Learning Res.*, vol. 5, pp. 1457–1469, 2004.
3. J. Shen and G. W. Israël, "A receptor model using a specific non-negative transformation technique for ambient aerosol," *Atmospheric Environment*, vol. 23, no. 10, pp. 2289–2298, 1989.
4. P. Paatero and U. Tapper, "Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values," *Environmetrics*, vol. 5, pp. 111–126, 1994.
5. W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," in *SIGIR '03: Proc. of the 26th Ann. Intl. ACM SIGIR Conf. on Res. and Developm. in Info. Retrieval*, pp. 267–273, ACM Press, 2003.
6. P. O. Hoyer and A. Hyvärinen, "A multi-layer sparse coding network learns contour coding from natural images," *Vision Research*, vol. 42, no. 12, pp. 1593–1605, 2002.
7. P. Smaragdis and J. C. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *IEEE Workshop on Appl. of Sign. Proc. to Audio and Acoustics*, pp. 177–180, 2003.
8. D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, Oct. 1999.

9. D. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts?," in *Adv. in NIPS*, vol. 17, 2004.
10. S. Z. Li, X. W. Hou, H. J. Zhang, and Q. S. Cheng, "Learning spatially localized, parts-based representation," in *Proc. of CVPR*, 2001.
11. Y. Wang, Y. Jia, C. Hu, and M. Turk, "Fisher non-negative matrix factorization for learning local features," in *Proc. Asian Conf. on Comp. Vision*, 2004.
12. N. Littlestone and M. Warmuth, "Relating data compression, learnability, and the Vapnik-Chervonenkis dimension," tech. rep., Univ. of Calif. Santa Cruz, 1986.
13. R. Herbrich and R. C. Williamson, "Algorithmic luckiness," *J. of Mach. Learning Res.*, vol. 3, pp. 175–212, 2002.
14. B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by V1?," *Vision Research*, vol. 37, pp. 3311–3325, Dec. 1997.
15. J. F. Sturm, *Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones (updated version 1.05).* Department of Econometrics, Tilburg University, Tilburg, The Netherlands, 2001.
16. M. ApS, ed., *The MOSEK optimization tools version 3.2 (Revision 8) User's manual and reference.* MOSEK ApS, Denmark, 2005.
17. M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Applications*, 1998.
18. H. Tuy, "Convex programs with an additional reverse convex constraint," *J. of Optim. Theory and Applic.*, vol. 52, pp. 463–486, Mar. 1987.
19. R. Horst and P. M. Pardalos, eds., *Handbook of Global Optimization.* Kluwer Academic Publisher, 1995.
20. A. d'Aspremont, L. E. Ghaoui, M. I. Jordan, and G. R. Lanckriet, "A direct formulation for sparse PCA using semidefinite programming," in *Adv. in NIPS*, 2004.
21. H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," (J. of Comp. a. Graph. Statistics, to appear).
22. C. Chennubholta and A. Jepson, "Sparse PCA extracting multi-scale structure from data," in *Proc. of ICCV*, pp. 641–647, 2001.
23. P. Paatero, "Least squares formulation of robust non-negative factor analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 37, 1997.
24. CBCL, "CBCL face database #1." MIT Center For Biological and Computational Learning, http://cbcl.mit.edu/software-datasets, 2000.