Fast Nonnegative Matrix Factorization Using Nested ADMM Iterations

Wu-Sheng Lu
Electical and Computer Engineering
University of Victoria, Victoria, BC, Canada
Email: ismailds@uvic.ca, wslu@uvic.ca

***Abstract*—As a constrained low-rank decomposition technique nonnegative matrix factorization (NMF) finds a wide variety of applications, especially in the analysis and design of pattern recognition systems for large-scale datasets. In this paper, we present a new algorithm for NMF based on nested alternating direction method of multipliers (ADMM) iterations. The paper describes the algorithm with a great deal of technical details, and includes a case study to demonstrate the algorithm's ability to handle large-scale datasets with improved efficiency in comparison with those not using nested ADMM iterations.**

I. Introduction

Nonnegative matrix factorization (NMF) is a dimensionality reduction technique which finds a wide variety of applications such as sparse representation of facial data, semantic analysis of text documents, and decomposition of hyperspectral images [1][2]. A matrix $\boldsymbol{M}$ is said to be nonnegative (and is denoted by $\boldsymbol{M} \geq \boldsymbol{0}$) if its components are all nonnegative. Mathematically, NMF may be regarded as a constrained low-rank decomposition where a nonnegative matrix $\boldsymbol{X} \in \boldsymbol{R}^{n\times m}$ is approximately represented by product $\boldsymbol{WH}$ where $\boldsymbol{W} \in \boldsymbol{R}^{n\times r}$ and $\boldsymbol{H} \in \boldsymbol{R}^{r\times m}$ with $r \ll \min\{n, m\}$, and both $\boldsymbol{W}$ and $\boldsymbol{H}$ are constrained to be nonnegative. The popularity of NMF gained since the pioneering work [1] has largely to do with its ability to extract sparse and interpretable features from a nonnegative data matrix [1][2]. Many algorithms for NMF in various formulations have been proposed over the last two decades [2]. In this paper, the NMF problem is addressed as a regularized least-squares problem subject to nonnegativity constraints. The contribution of the paper is the development of an alternating convex optimization algorithm where (i) each round solves two convex problems each of which is decomposed into a number of convex subproblems of much reduced size that can be solved in parallel (hence fast); (ii) each subproblem is solved inexactly (hence fast) using a small number of alternating direction method of multipliers (ADMM) iterations [3]; and (iii) since the minimization required in one of the ADMM steps is nontrivial to solve, a secondary set of ADMM iterations is embedded into it to address the matter, so overall the algorithm's has a structure that may be characterized as one with nested ADMM iterations. The proposed algorithm is shown to be able to handle large-scale datasets with improved efficiency in comparison with those that do not employ nested ADMM iterations.

II. Problem Formulation

In this paper, the NMF problem is addressed by solving the regularized constrained minimization problem

$$\underset{\boldsymbol{W},\boldsymbol{H}}{\text{minimize}} \quad \tfrac{1}{2}\|\boldsymbol{WH}-\boldsymbol{X}\|_F^2 + \mu_1 \sum_{i=1}^{n}\sum_{j=1}^{r}|w_{i,j}| + \mu_2 \sum_{i=1}^{r}\sum_{j=1}^{m}|h_{i,j}| \tag{1a}$$

$$\text{subject to:} \quad \boldsymbol{W} \geq \boldsymbol{0},\ \boldsymbol{H} \geq \boldsymbol{0} \tag{1b}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, $w_{i,j}$ and $h_{i,j}$ are the (i,j)th components of $\boldsymbol{W}$ and $\boldsymbol{H}$, respectively, μ_1 and μ_2 are positive constants, known as regularization parameters, that are selected so that the associated terms promote the sparsity of matrices $\boldsymbol{W}$ and $\boldsymbol{H}$, respectively. It has been shown [1] that for a facial data matrix $\boldsymbol{X}$, there exist nonnegative matrices $\boldsymbol{W}$ and $\boldsymbol{H}$ with a small r such that product $\boldsymbol{WH}$ provides a good approximation of $\boldsymbol{X}$, where the columns of $\boldsymbol{W}$, often regarded as basis vectors, are visually interpretable as parts of human faces such as eyes, noses, lips, etc., whereas the components of a column of matrix $\boldsymbol{H}$ are nonnegative weights that linearly combine the basis vectors and add up to reconstruct the corresponding facial image of input data matrix $\boldsymbol{X}$.

It is important to note that because of the presence of product $\boldsymbol{WH}$ in the Frobenius norm, (1) is a *nonconvex* problem and hence is expected to admit multiple local solutions. Another technical challenge is that the problem contains a total of $n\cdot r+r\cdot m$ variables, suggesting that (1) can be a *large-scale* problem. For the database from the Center for Biological and Computational Learning (CBCL) at MIT [4], for example, one deals with a data matrix $\boldsymbol{X}$ of size 361×2429, where each column is generated by stacking a facial image of 19×19 pixels. If we set $r = 49$ as many researchers do, problem (1) in this case contains a total of $(n+m)\cdot r = 136,710$, variables.

III. The Algrithm

The algorithm is developed in an alternating convex optimization framework and contains several algorithmic modules, each involves ADMM iterations.

A. An Alternating convex optimization framework

To deal with this nonconvex and possibly large-scale problem, an alternating convex optimization framework is employed. It works as follows: First, matrix $\boldsymbol{H}$ is held fixed while solving problem (1) with respect to matrix $\boldsymbol{W}$; next,

978-1-7281-2794-1/19/$31.00 ©2019 IEEE

matrix $\boldsymbol{W}$ is held fixed to the solution just obtained while solving (1) with respect to matrix $\boldsymbol{H}$. This pair of alternating optimizations is repeated a prescribed number N of times, or until the objective function in (1a) reaches a steady state up to a convergence tolerance ε. The convergence of this process is guaranteed because the process always yields a monotonically decreasing profile of the objective function, and the profile is bounded by the zero value from below (because the objective function is always nonnegative). It is important to note that the sub-problems that needs to be solved in the alternating procedure described above are standard $L_1 - L_2$ problems and hence convex [5].

B. Problem Decomposition

The sizes of the sub-problems involved are $r \cdot m$ (when $\boldsymbol{W}$ is fixed) and $n \cdot r$ (when $\boldsymbol{H}$ is fixed) which can still be quite large. In the case of CBCL database, for example, the two sub-problems involve 119,021 and 17,689 variables, respectively. To address the issue, we let

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w}_1^T \\ \boldsymbol{w}_2^T \\ \vdots \\ \boldsymbol{w}_n^T \end{bmatrix}, \ \boldsymbol{H} = [\, \boldsymbol{h}_1 \quad \boldsymbol{h}_2 \quad \cdots \quad \boldsymbol{h}_m \,],$$

$$\boldsymbol{X} = [\, \boldsymbol{x}_1 \quad \boldsymbol{x}_2 \quad \cdots \quad \boldsymbol{x}_m \,], \ = \begin{bmatrix} \hat{\boldsymbol{x}}_1^T \\ \hat{\boldsymbol{x}}_2^T \\ \vdots \\ \hat{\boldsymbol{x}}_n^T \end{bmatrix}$$

and note that in the sub-problem where $\boldsymbol{H}$ is held fixed, the third term of the objective function in (1a) becomes constant hence can be neglected, and the remaining terms can be expressed as

$$\begin{aligned} &\tfrac{1}{2}\|\boldsymbol{W}\boldsymbol{H} - \boldsymbol{X}\|_F^2 + \mu_1 \sum_{i=1}^{n}\sum_{j=1}^{r} |w_{i,j}| \\ &= \sum_{i=1}^{n} \left[\tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w}_i - \hat{\boldsymbol{x}}_i\|_2^2 + \mu_1 \|\boldsymbol{w}_i\|_1 \right] \end{aligned}$$

where each of the variable vectors $\{\boldsymbol{w}_i, i = 1,\ 2,\ \ldots,\ n\}$ appears only in one of the n-term sums at the right-hand side of the above expression. As a result, the minimization problem at hand can be decomposed into n small-size problems, each deals with one of the variable vectors, $\boldsymbol{w}_i$, by solving a standard $L_1 - L_2$ minimization problem subject to nonnegativity of $\boldsymbol{w}_i$:

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w}_i - \hat{\boldsymbol{x}}_i\|_2^2 + \mu_1\|\boldsymbol{w}_i\|_1 && \text{(2a)} \\ \text{subject to:} \quad & \boldsymbol{w}_i \geq 0 && \text{(2b)} \end{aligned}$$

which is a convex problem involving only r variables. Similarly, in the sub-problem where $\boldsymbol{W}$ is held fixed, the second term of the objective function in (1a) becomes constant hence can be neglected, and the remaining terms can be expressed as

$$\begin{aligned} &\tfrac{1}{2}\|\boldsymbol{W}\boldsymbol{H} - \boldsymbol{X}\|_F^2 + \mu_2 \sum_{i=1}^{r}\sum_{j=1}^{m} |h_{i,j}| \\ &= \sum_{i=1}^{m} \left[\tfrac{1}{2}\|\boldsymbol{W}\boldsymbol{h}_j - \hat{\boldsymbol{x}}_j\|_2^2 + \mu_2 \|\boldsymbol{h}_j\|_1 \right] \end{aligned}$$

where each of the variable vectors $\{\boldsymbol{h}_j,\ j = 1,\ 2,\ \ldots,\ m\}$ appears only in one of the m-term sums at the right-hand side of the above expression. Consequently, the minimization of the objective function can be performed by optimizing each $\boldsymbol{h}_j$ that can be carried out by solving a standard $L_1 - L_2$ minimization problem subject to nonnegativity of $\boldsymbol{h}_j$, namely,

$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\boldsymbol{W}^T\boldsymbol{h}_j - \boldsymbol{x}_j\|_2^2 + \mu_2\|\boldsymbol{h}_j\|_1 && \text{(3a)} \\ \text{subject to:} \quad & \boldsymbol{h}_j \geq 0 && \text{(3b)} \end{aligned}$$

which is a convex problem involving only r variables. We remark that for applications where fast algorithm implementation is a top priority, the small-size problems in (2) and (3) can be carried out in parallel.

C. Solving Problems (2) and (3) with Nested ADMM

The standard ADMM iterations are aimed at solving the class of convex problems

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) + h(\boldsymbol{y}) && \text{(4a)} \\ \text{subject to:} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y} = \boldsymbol{c} && \text{(4b)} \end{aligned}$$

where $f(\boldsymbol{x})$ and $h(\boldsymbol{y})$ are convex and $\{\boldsymbol{A},\ \boldsymbol{B},\ \boldsymbol{c}\}$ are known matrices of appropriate dimensions. The scaled ADMM iterations for problem (4) are given by

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \arg\min_{x} \left[f(\boldsymbol{x}) + \tfrac{\alpha}{2}\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{y}_k - \boldsymbol{c} + \boldsymbol{\nu}_k\|_2^2 \right] && \text{(5a)} \\ \boldsymbol{y}_{k+1} &= \arg\min_{y} \left[h(\boldsymbol{y}) + \tfrac{\alpha}{2}\|\boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{B}\boldsymbol{y} - \boldsymbol{c} + \boldsymbol{\nu}_k\|_2^2 \right] && \text{(5b)} \\ \boldsymbol{\nu}_{k+1} &= \boldsymbol{\nu}_k + \boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{B}\boldsymbol{y}_{k+1} - \boldsymbol{c} && \text{(5c)} \end{aligned}$$

where $\alpha > 0$ is a parameter to control the quadratic penalty terms in (5a) and (5b) for violation of the equality constraint in (4b), and $\boldsymbol{\nu}_k$ is a Lagrange multiplier associated with (4b).

To illustrate how ADMM iterations can be applied to problems (2) and (3), we define

$$\begin{aligned} f(\boldsymbol{w}) &= \tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w} - \hat{\boldsymbol{x}}\|_2^2 + \mu_1\|\boldsymbol{w}\|_1 \\ I_c(\boldsymbol{y}) &= \begin{cases} 0 & \text{if } \boldsymbol{y} \in \mathcal{C} \\ +\infty & \text{if } \boldsymbol{y} \notin \mathcal{C} \end{cases} \end{aligned}$$

and $\mathcal{C} = \{\boldsymbol{y} : \boldsymbol{y} \geq \boldsymbol{0}\}$, and express problem (2) as

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{w}_i) + I_c(\boldsymbol{y}) && \text{(6a)} \\ \text{subject to:} \quad & \boldsymbol{w}_i - \boldsymbol{y} = \boldsymbol{0} && \text{(6b)} \end{aligned}$$

which fits nicely into the formulation in (4) with $\boldsymbol{x} = \boldsymbol{w}_i$, $\boldsymbol{A} = \boldsymbol{I}$, $\boldsymbol{B} = -\boldsymbol{I}$, and $\boldsymbol{c} = \boldsymbol{0}$. Consequently, the ADMM

iterations in (5) are applicable to problem (2) and, with necessary notation changes, we have

$$\begin{aligned}
\boldsymbol{w}_{k+1} &= \arg\min_{w} \Big[\tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w}-\hat{\boldsymbol{x}}\|_2^2 \\
&\quad +\tfrac{\alpha}{2}\,\|\boldsymbol{w}-(\boldsymbol{y}_k-\boldsymbol{\nu}_k)\|_2^2+\mu_1\|\boldsymbol{w}\|_1\Big] \qquad (7a)\\
\boldsymbol{y}_{k+1} &= \arg\min_{y}\big[I_c(\boldsymbol{y})+\tfrac{\alpha}{2}\|\boldsymbol{y}-(\boldsymbol{w}_{k+1}+\boldsymbol{\nu}_k)\|_2^2\big] \qquad (7b)\\
\boldsymbol{\nu}_{k+1} &= \boldsymbol{\nu}_k+\boldsymbol{w}_{k+1}-\boldsymbol{y}_{k+1} \qquad (7c)
\end{aligned}$$

for $k = 0,\ 1,\ \ldots,\ K$. The problem in (7b) involves an indicator function for set $\mathcal{C}$, as a result its solution is simply the componentwise projection of vector $\boldsymbol{w}_{k+1}+\boldsymbol{\nu}_k$ onto the nonnegative real axis, namely,

$$\boldsymbol{y}_{k+1} = P_c(\boldsymbol{w}_{k+1}+\boldsymbol{\nu}_k) = \max(\boldsymbol{w}_{k+1}+\boldsymbol{\nu}_k, \boldsymbol{0}) \qquad (8)$$

The problem in (7a) is essentially an L_1-L_2 problem which is convex and hence several options are available for solving it. One of the options is a technique known as soft-shrinkage (or soft-thresholding) [6], where the amount of shrinkage made in each iteration is inversely proportional to Lipschitz constant L of the objective functions gradient. Unfortunately, in many applications involving large-scale data, L is usually very large, leading to slow convergence. To deal with this issue, in our algorithm problem (7a) itself is solved by ADMM iterations. To this end, (7a) is formulated as

$$\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{w})+\mu_1\|\boldsymbol{z}\|_1\\
\text{subject to:} \quad & \boldsymbol{w}-\boldsymbol{z}=\boldsymbol{0}
\end{aligned}$$

to fit it into an ADMM framework, where

$$f(\boldsymbol{w}) = \tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w}-\hat{\boldsymbol{x}}\|_2^2+\tfrac{\alpha}{2}\|\boldsymbol{w}-(\boldsymbol{y}_k-\boldsymbol{\nu}_k)\|_3^2$$

By applying scaled ADMM iterations to the above problem, we obtain

$$\begin{aligned}
\boldsymbol{w}_{l+1} &= \arg\min_{w}\Big[\tfrac{1}{2}\|\boldsymbol{H}^T\boldsymbol{w}-\hat{\boldsymbol{x}}\|_2^2+\tfrac{\alpha}{2}\|\boldsymbol{w}-(\boldsymbol{y}_k-\boldsymbol{\nu}_k)\|_2^2\\
&\quad +\tfrac{\beta}{2}\|\boldsymbol{w}-(\boldsymbol{z}_l-\boldsymbol{\lambda}_l)\|_2^2\Big] \qquad (9a)\\
\boldsymbol{z}_{l+1} &= \arg\min_{z}\big[\mu_1\|\boldsymbol{z}\|_1+\tfrac{\beta}{2}\|\boldsymbol{z}-(\boldsymbol{w}_{l+1}+\boldsymbol{\lambda}_l)\|_2^2\big] \qquad (9b)\\
\boldsymbol{\lambda}_{l+1} &= \boldsymbol{\lambda}_k+\boldsymbol{w}_{l+1}-\boldsymbol{z}_{l+1} \qquad (9c)
\end{aligned}$$

for $l = 0,\ 1,\ \ldots,\ M$ Solving problem (9) is rather straightforward: (9a) is a simple convex quadratic problem which admits a global solution in closed-form as

$$\boldsymbol{w}_{l+1} = \Big(\boldsymbol{H}\boldsymbol{H}^T+(\alpha+\beta)\boldsymbol{I}\Big)^{-1}\boldsymbol{b}_l \qquad (10a)$$

where

$$\boldsymbol{b}_l = \boldsymbol{H}\hat{\boldsymbol{x}}+\alpha(\boldsymbol{y}_k-\boldsymbol{\nu}_k)+\beta(\boldsymbol{z}_l-\boldsymbol{\lambda}_l) \qquad (10b)$$

whereas (9b) can be solved by standard soft-shrinkage in closed-form as

$$\boldsymbol{z}_{l+1} = \text{sgn}(\boldsymbol{c}_l)\cdot\max\left\{|\boldsymbol{c}_l|-\frac{\mu_1}{\beta},\ \boldsymbol{0}\right\} \qquad (11a)$$

where

$$\boldsymbol{c}_l = \boldsymbol{w}_{l+1}+\boldsymbol{\lambda}_l \qquad (11b)$$

It is important to note that the matrix inverse required by (10a) only needs to be evaluated once in the entire solution process because the inverse does not depend on k and hence it can be used in every iteration. The result of the above analysis is an algorithmic procedure to solve the problem in (2) where two sets of ADMM iterations, with one nested in another, are utilized. A similar procedure can be used to solve the problem in (3) and hence to solve the NMF problem in (1).

D. Summary of the algorithm

The algorithm proposed above is summarized as follows.

Input: Data matrix $\boldsymbol{X}$; parameters r, μ_1, μ_2, α, β, N, K, and M; and initial $\boldsymbol{H}_0$, $\boldsymbol{y}_0$, $\boldsymbol{z}_0$, $\boldsymbol{\nu}_0$ and $\boldsymbol{\lambda}_0$.

for $0,\ 1,\ \ldots,\ N-1$, do

- Compute each row of $\boldsymbol{W}_{n+1}$ by running K rounds of ADMM iterations (7) where $\boldsymbol{H}$ is set to $\boldsymbol{H}_n$ and $\hat{\boldsymbol{x}}$ is set to the transpose of the corresponding row from $\boldsymbol{X}$; To solve (7a), run M rounds of ADMM iterations (9) where (9a) is solved using (10) and (9b) is solved using (11).
- Compute each column of $\boldsymbol{H}_{m+1}$ by running K rounds of ADMM iterations (7) where $\boldsymbol{H}^T$ is replaced by $\boldsymbol{W}_{n+1}$ and $\hat{\boldsymbol{x}}$ is replaced by the corresponding column from $\boldsymbol{X}$; To solve (7a), run M rounds of ADMM iterations (9) where (9a) is solved using (10) and (9b) is solved using (11).

end

Denote the last pair of matrices $\{\boldsymbol{W}_N, \boldsymbol{H}_N\}$ obtained from the algorithm by $\{\boldsymbol{W}^*, \boldsymbol{H}^*\}$ and take it to be a solution of problem (1).

IV. A Case Study

The algorithm proposed above was evaluated by applying it to the CBCL database of facial images (see Sec. II), where matrix $\boldsymbol{X}$ is of size 361×2429. Dimension r was set to 49 and the regularization parameters were set to $\mu_1 = 0.015$ and $\mu_2 = 1$. Other parameters were set to $\alpha=\beta=0.05$, $N=200$, $K=M=35$, and a random nonnegative matrix of size 49×2429 was used as initial $\boldsymbol{H}_0$ to start the algorithm. The proposed algorithm yielded a solution $\{\boldsymbol{W}^*, \boldsymbol{H}^*\}$ and hence a nonnegative rank-r (with $r=49$) approximation of data set $\boldsymbol{X}$ as $\boldsymbol{X}\approx\boldsymbol{W}^*\boldsymbol{H}^*$. The relative approximation error in Frobenius norm was found to be

$$e_{rf} = \frac{\|\boldsymbol{X}-\boldsymbol{W}^*\boldsymbol{H}^*\|_F}{\|\boldsymbol{X}\|_F} = 0.0816$$

For comparison, the rank-49 approximation of $\boldsymbol{X}$ based on singular value decomposition (SVD), which is known to reach the global minimum e_{rf} [7], offers an $e_{rf}=0.0752$. In view of the closeness of the e_{rf} achieved by our solution to the global low bound and the fact that in general the SVD-based approximation is not an NMF, $\{\boldsymbol{W}^*, \boldsymbol{H}^*\}$ obtained by the proposed algorithm is considered an excellent local solution.

Because of the inclusion of the two regularization terms in (1a), both $\boldsymbol{W}^*$ and $\boldsymbol{H}^*$ are sparse. Fig. 1 shows 49 images, each was obtained by converting a column of matrix $\boldsymbol{W}^*$ into a square matrix of 19×19 gray-scale pixels. We see

that these matrices are not only sparse, they are *interpretable* as various "parts" of human faces, thus can be utilized as basis images for reconstructing the facial images from input dataset $\boldsymbol{X}$. Fig. 2 displays 49 components of a typical column of $\boldsymbol{H}^*$. The reconstruction of the jth input facial image $\boldsymbol{x}_j$ can readily be obtained by using the r components of the jth column of matrix $\boldsymbol{H}^*$ as weights to linearly combine the r basis images from $\boldsymbol{W}^*$. The sparsity and nonnegativity of $\boldsymbol{H}^*$ imply that a facial image can be well approximated by selecting a small number of basic parts and adding them up with appropriate weights. For illustration, Figs. 3(a) and 3(b) depict the first 49 input facial images and their approximations by NMF, respectively.

As mentioned earlier, (1) is a nonconvex problem admitting multiple local solutions, and the selection of an initial point, which is the initial matrix $\boldsymbol{H}_0$, effects the performance of the solution point. Depicted in Fig. 4 are eight profiles of the objective function versus the first 40 rounds of iterations that were generated by the algorithm when it started off from eight random $\boldsymbol{H}_0$ produced using different seeds. It is observed that different $\boldsymbol{H}_0$'s do lead to different (local) solutions, but all eight solutions offer comparable performance.

The computational complexity of the proposed algorithm was evaluated by comparing with its counterparts in two scenarios: one replaces the second set of ADMM iterations (see (9)–(11)) by a standard interior-point convex minimization to solve (7a). Relative to the CPU time required by the proposed algorithm, which was normalized to one unit, the algorithm with the above changes required 1.2921 units of normalized CPU time to obtain a local solution with comparable performance. If in addition one also replaces the first set of ADMM iterations (see (7)–(8)) by a standard interior-point convex optimization, the normalized CPU time required was increased to 1.4937 units.

V. Conclusions

We have proposed a new algorithm for the NMF problem. The algorithm is developed in an alternating convex optimization framework and two sets of ADMM iterations are used to solve the convex sub-problems involved. A case study is presented to evaluate the performance of the algorithm and demonstrate its ability to deal with large-scale datasets.

References

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

[2] N. Gillis, "The why and how of nonnegative matrix factorization," in *Regularization, Optimization, Kernels, and Support Vector Machines*, J. Suykens, M. Signoretto, and A. Argyriou eds., pp. 257–291. Chapman & Hall/CRC, 2014.

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[4] Pattern classification database of the Center for Biological and Computational Learning at MIT. [Online]. Available: https://github.com/HyTruongSon/Pattern-Classification/tree/master/MIT-CBCL-database

[5] A. Antoniou and W.-S. Lu, *Practical Optimization: Algorithms and Engineering Applications*, Springer, 2007.

[6] D. Donoho, "De-noising by soft thresholding," *IEEE Trans. Information Theory*, vol. 41, pp. 613–627, 1995.

[7] G. W. Stewart, *Introduction to Matrix Computations,* Academic Press, 1973.

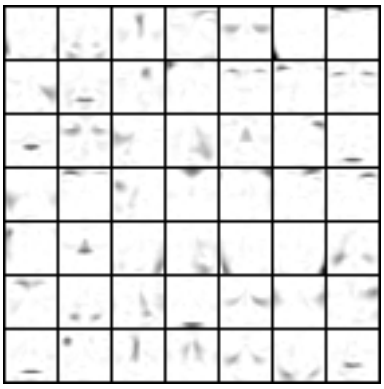

Fig. 1. 49 columns of $\boldsymbol{W}^*$, each column was converted into an images of 19×19 pixels.

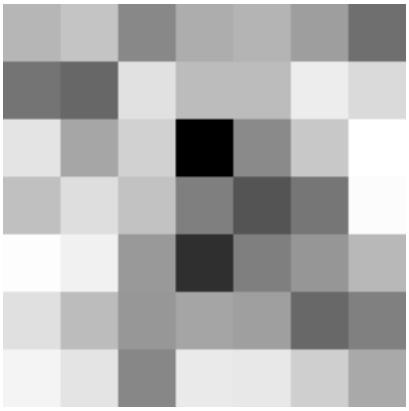

Fig. 2. A typical column of $\boldsymbol{H}^*$ converted to an image of 7×7 pixels. Lighter blocks represent smaller numerical values.

Fig. 3. (a) First 49 facial images from CBCL database.

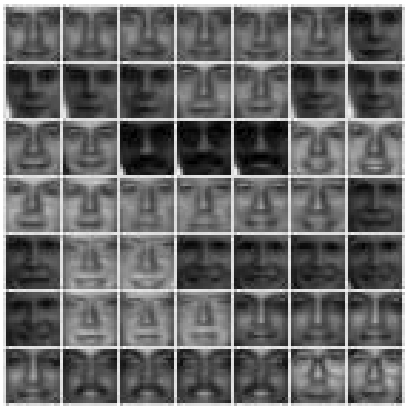

Fig. 3. (b) Images produced by the first 49 columns of $\boldsymbol{W}^*\boldsymbol{H}^*$.

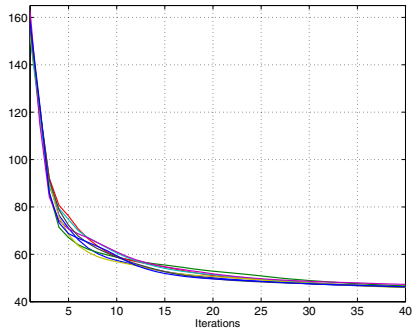

Fig. 4. Profiles of the objective function in (1a) over first 40 iterations from eight random initial $\boldsymbol{H}_0$.