Fast Projection-Based Methods for the Least Squares Nonnegative Matrix Approximation Problem

Dongmin Kim, Suvrit Sra, Inderjit S. Dhillon
Department of Computer Sciences, University of Texas Austin, TX 78712-1188, USA

Received 16 July 2007; accepted 7 August 2007
DOI:10.1002/sam.104
Published online 28 December 2007 in Wiley InterScience (www.interscience.wiley.com).

Abstract: Nonnegative matrix approximation (NNMA) is a popular matrix decomposition technique that has proven to be useful across a diverse variety of fields with applications ranging from document analysis and image processing to bioinformatics and signal processing. Over the years, several algorithms for NNMA have been proposed, e.g. Lee and Seung's multiplicative updates, alternating least squares (ALS), and gradient descent-based procedures. However, most of these procedures suffer from either slow convergence, numerical instability, or at worst, serious theoretical drawbacks. In this paper, we develop a new and improved algorithmic framework for the least-squares NNMA problem, which is not only theoretically well-founded, but also overcomes many deficiencies of other methods. Our framework readily admits powerful optimization techniques and as concrete realizations we present implementations based on the Newton, BFGS and conjugate gradient methods. Our algorithms provide numerical results superior to both Lee and Seung's method as well as to the alternating least squares heuristic, which was reported to work well in some situations but has no theoretical guarantees [1]. Our approach extends naturally to include regularization and box-constraints without sacrificing convergence guarantees. We present experimental results on both synthetic and real-world datasets that demonstrate the superiority of our methods, both in terms of better approximations as well as computational efficiency. © 2007 Wiley Periodicals, Inc. Statistical Analy Data Mining 1: 38–51, 2008

Keywords: nonnegative matrix approximation; factorization; projected Newton methods; active sets; least-squares

1. INTRODUCTION

Nonnegative matrix approximation (NNMA), also known as *nonnegative matrix factorization* [2] or *positive matrix factorization* [3], is a popular and effective matrix decomposition technique. It has become an established method for performing dimensionality reduction and related tasks such as clustering, image processing, and visualization—with applications across a diverse variety of fields. The NNMA problem setting is defined as follows. Let $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N]$ be the matrix of nonnegative inputs, where each column $\boldsymbol{a}_i \in \mathbb{R}^M_+$. NNMA seeks to approximate these input vectors by nonnegative linear (conic) combinations of a small number of *nonnegative representative vectors*, $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_K\}$, so that

$$\boldsymbol{a}_i \approx \sum_{k=1}^{K} \boldsymbol{b}_k c_{ki}, \tag{1}$$

where the coefficients c_{ki} are also nonnegative. We remark in passing that various alternative restrictions on $\boldsymbol{b}_k$ or $\boldsymbol{c}_i = [c_{1i}\ c_{2i}\ \ldots\ c_{Ki}]^T$ may be placed to obtain different types of approximations. For the purpose of this paper, we focus only on the problem with nonnegativity constraints.

The quality of the approximation in Eq. (1) may be measured using an appropriate distortion function, for example, the Frobenius norm distortion or the Kullback-Leibler divergence. In this paper we focus on the former distortion, which leads to the following *least-squares NNMA* problem:

$$\underset{\boldsymbol{B},\boldsymbol{C}\geq 0}{\text{minimize}}\ \mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = \frac{1}{2}\|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}^2, \tag{2}$$

where $\boldsymbol{A}$ is the input matrix and $\boldsymbol{B}, \boldsymbol{C}$ are the output (factor) matrices. The matrix $\boldsymbol{B}$ may be intuitively viewed as a set of basis vectors that can be conically combined using the coefficients in $\boldsymbol{C}$ to approximate the input $\boldsymbol{A}$.

In this paper we develop two new Newton-type algorithms for solving Eq. (2) along with a theoretical analysis establishing their convergence. Both our algorithms improve upon the *de facto* procedure of Lee and Seung [4], hereafter referred to as *LS*, as well as upon the popular ALS heuristic, which has been reported to perform well in practice [1]. However, both LS and ALS have their drawbacks;

Correspondence to: Suvrit Sra (suvrit@cs.utexas.edu)

the former suffers from slow convergence, whereas the latter lacks theoretical guarantees on its performance—our new algorithms rectify both of these deficiencies.

Researchers have also considered the following regularized NNMA problem

$$\underset{\boldsymbol{B},\boldsymbol{C}\geq 0}{\text{minimize}} \quad \frac{1}{2}\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}^2+\lambda\|\boldsymbol{B}\|_{\mathrm{F}}^2+\mu\|\boldsymbol{C}\|_{\mathrm{F}}^2, \qquad (3)$$

where $\lambda > 0$, and $\mu > 0$ are regularization parameters. The motivation behind studying Eq. (3) can be ascribed to certain practical concerns. For example, the basic NNMA problem estimates the product $\boldsymbol{BC}$ that has $(M+N)K$ parameters. Such a large number of parameters can lead to overfitting, which despite the apparent sparse representations yielded by NNMA, might be difficult to counter without regularization. Furthermore, the regularization terms also make the optimization problem more numerically stable.

Another interesting variation arises when one binds the solution values by imposing box-constraints on the variables. For NNMA this results in the problem

$$\begin{array}{ll}\text{minimize} & \frac{1}{2}\|\boldsymbol{A}-\boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}^2,\\ \text{subject to} & \boldsymbol{P}\leq\boldsymbol{B}\leq\boldsymbol{Q},\\ & \boldsymbol{R}\leq\boldsymbol{C}\leq\boldsymbol{S},\end{array} \qquad (4)$$

where the inequalities are component-wise. Both Eqs. (3) and (4) can be handled by our methods without much additional difficulty.

2. BACKGROUND AND RELATED WORK

The NNMA objective function Eq. (2) is not simultaneously convex in both $\boldsymbol{B}$ and $\boldsymbol{C}$ due to the presence of the product term $\boldsymbol{BC}$. Hence, in general it is very difficult to find globally optimal solutions to Eq. (2). Fortunately, the objective function is at least individually convex in $\boldsymbol{B}$ and in $\boldsymbol{C}$, which makes it possible to invoke an alternating minimization or descent procedure that takes the form:

1. Initialize $\boldsymbol{B}^0$ and/or $\boldsymbol{C}^0$; set $t \leftarrow 0$.
2. Fix $\boldsymbol{B}^t$ and find $\boldsymbol{C}^{t+1}$ such that
$$\mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C}^{t+1})\leq\mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C}^t).$$
3. Fix $\boldsymbol{C}^{t+1}$ and find $\boldsymbol{B}^{t+1}$ such that
$$\mathcal{F}(\boldsymbol{B}^{t+1},\boldsymbol{C}^{t+1})\leq\mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C}^{t+1}).$$
4. Let $t \leftarrow t+1$, and repeat Steps 2 and 3 until convergence.

On the basis of the above procedure, we can categorize NNMA methods into two types, namely the *exact* and *inexact* methods. The former perform an exact minimization at each iterative step so that $\boldsymbol{C}^{t+1} = \operatorname{argmin}_{\boldsymbol{C}} \mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C})$ (similarly for $\boldsymbol{B}^{t+1}$), while the latter merely ensure descent, i.e. $\mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C}^{t+1}) \leq \mathcal{F}(\boldsymbol{B}^t,\boldsymbol{C}^t)$ (similarly for $\boldsymbol{B}^{t+1}$).

Since the Frobenius norm of a matrix is just the sum of Euclidean norms over columns (or rows), minimization or descent over either $\boldsymbol{B}$ or $\boldsymbol{C}$ boils down to solving a sequence of nonnegative least squares (NNLS) problems of the form

$$\begin{array}{ll}\underset{x}{\text{minimize}} & f(\boldsymbol{x})=\frac{1}{2}\|\boldsymbol{G}\boldsymbol{x}-\boldsymbol{h}\|_2^2,\\ \text{subject to} & \boldsymbol{x}\geq\boldsymbol{0}.\end{array} \qquad (5)$$

Exact methods find a global optimum of this subproblem, while inexact methods roughly approximate it. There do exist well-known methods for solving the NNLS problem, such as the Lawson-Hanson procedure [5], FNNLS [6], and other procedures mentioned in ref. [7]. However, as we show in ref. [8], our approach to solving NNLS outperforms the other methods, hence we favor it as the method of choice for solving Eq. (5). At this point, we alert the readers against a potential misinterpretation that could arise from our choice of nomenclature in terms of exact and inexact methods. It is not the case that the exact methods are superior to the inexact ones, or even that the exact methods could converge to a global optimum of Eq. (2). However, the exact methods do provide better theoretical properties and they tend to produce better quality solutions, even-though there is still no guarantee on the global optimality due to the nonconvexity of Eq. (2). Inexact methods often provide great savings of computational effort by trading-off precision of the solutions for speed.

In this paper we present a new exact method for NNMA, which we call FNMA$^{\mathrm{E}}$. There have been other exact approaches in the literature. For example, Paatero and Tapper [3, 9, 10] introduced a set of algorithms for NNMA and provided a convergence proof for *one* of their methods that employs the preconditioned conjugate gradient method. However, their methods are described in a nebulous fashion, and they cite the need for considerable engineering effort [9] for an actual implementation. Bierlaire *et al.* [11] developed a projected gradient method for NNLS, which Lin [12] applied to solve problem Eq. (2). Recently, Merritt and Zhang [13] developed an interior-point gradient method for NNLS—a gradient descent-based method without projection that maintains feasibility of intermediate solutions throughout the iterations. They also provided a convergence proof for their method under the mild assumption that $\boldsymbol{G}$ has full-rank. Though problem Eq. (5) can be solved by any constrained optimization technique, the above methods are all based on gradient descent since it

allows for efficient handling of simple nonnegativity constraints. However, gradient-based methods are known to have linear convergence rate at best, and often suffer from a phenomenon known as *zigzagging or jamming*. FNMA$^{\mathrm{E}}$ subsumes the projected gradient-based method as a special case and retains its algorithmic simplicity while overcoming its deficiencies by employing a nondiagonal gradient scaling matrix.

The group of *inexact* methods has witnessed greater popularity and it includes Lee and Seung's [4] multiplicative algorithms. Gonzalez and Zhang [14] proposed a variant of Lee and Seung's method that utilizes a different scaling scheme for negative gradients to get faster convergence. Berry *et al*. [1] report the alternating least squares (ALS) procedure to be a simple but effective method for performing NNMA. The ALS procedure is somewhat *ad-hoc*—it solves the unconstrained least squares problem at each step exactly, followed by a truncation of the negative entries to zero. However, ALS does not have any convergence guarantees, and we discuss this in more detail in Section 2.1. Another inexact approach is provided by Zdunek and Cichocki's, which we refer to as the *ZC* method, Zdunek and Cichocki [15] who proposed the combination of projection with a quasi-Newton procedure for NNMA.

Our FNMA$^{\mathrm{E}}$ procedure is a quasi-Newton method that remedies the theoretical deficiencies of both the ALS and ZC methods. As an alternative, we also present an inexact method called FNMA$^{\mathrm{I}}$ that shares the same algorithmic framework as its exact counterpart FNMA$^{\mathrm{E}}$ while providing a computationally more efficient procedure.

2.1. ALS and ZC Methods

As alluded to the above, both the ALS and ZC methods have theoretical deficiencies, which can lead to nonmonotonic changes in the objective function value and to inferior approximations. We illustrate these deficiencies more clearly in this section providing further motivation for our algorithms.

Both ALS and ZC are closely related to FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$. A critical difference between FNMA$^{\mathrm{E}}$ and both these approaches (ZC and ALS) is that the former is an *exact* approach, whereas the latter two are *inexact* methods. To see why these methods are inexact, consider the NNLS subproblem Eq. (5) that they must solve. Let us denote a projection onto the nonnegative orthant by $\mathcal{P}_+[\cdot]$. Assuming $\boldsymbol{G}$ to be of full rank, the ALS update for subproblem Eq. (5) may be written as

$$\boldsymbol{x} = \mathcal{P}_+[(\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{h}], \tag{6}$$

or equivalently,

$$\boldsymbol{x} = \mathcal{P}_+[\boldsymbol{x} - (\boldsymbol{G}^T\boldsymbol{G})^{-1}(\boldsymbol{G}^T\boldsymbol{G}\boldsymbol{x} - \boldsymbol{G}^T\boldsymbol{h})].$$

For the ZC approach, the update is

$$\boldsymbol{x}^{\mathrm{new}} = \mathcal{P}_+[\boldsymbol{x}^{\mathrm{old}} - \alpha\boldsymbol{D}(\boldsymbol{G}^T\boldsymbol{G}\boldsymbol{x}^{\mathrm{old}} - \boldsymbol{G}^T\boldsymbol{h})], \tag{7}$$

where $\alpha > 0$ and $\boldsymbol{D}$ is some positive definite matrix that approximates $(\boldsymbol{G}^T\boldsymbol{G})^{-1}$, i.e. the inverse of the Hessian. Note that the ALS update has $\boldsymbol{D} = (\boldsymbol{G}^T\boldsymbol{G})^{-1}$ and $\alpha = 1$ in this form. Figure 1 illustrates why the updates Eqs. (6) and (7) are inexact, moreover they fail to decrease the objective function for an arbitrary positive α. Observe that Eq. (6) performs an exact-Newton step followed by projection, while Eq. (7) does a quasi-Newton with projection. Hence, we see that both the ALS and the ZC approaches can lead to an increase in the objective function value (also see Fig. 6). Our exact method, FNMA$^{\mathrm{E}}$, fixes this problem and is provably convergent unlike the ALS and ZC methods.

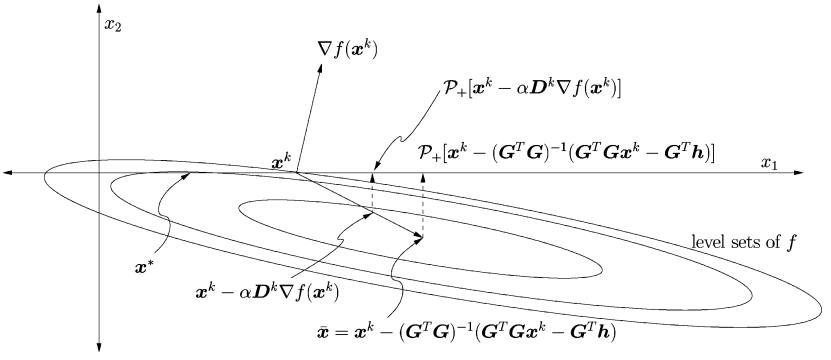

Fig. 1 Example where $\mathcal{P}_+[\boldsymbol{x}^k - \alpha\boldsymbol{D}^k\nabla f(\boldsymbol{x}^k)]$ fails to decrease the objective for an arbitrary $\alpha > 0$. In this figure, the ellipses represent level sets of f (the inner ellipses correspond to a smaller objective value), and $\boldsymbol{D}^k$ is assumed to be exactly equal to the inverse of the Hessian. The current iterate is given as $\boldsymbol{x}^k$. Note that for Problem Eq. (5), the Newton method reaches the unconstrained optimum $\bar{\boldsymbol{x}}$ in a single iteration. However, the projected solution $\mathcal{P}_+[\boldsymbol{x}^k - \boldsymbol{D}^k\nabla f(\boldsymbol{x}^k)]$ for nonnegatively constrained problems leads to an increase in the objective since the current iterate ($\boldsymbol{x}^k$) moves from an inner ellipse to an outer one by the update rule.

3. ALGORITHMS AND THEORY

In this section, we develop an algorithm and associated supporting theory for solving Eq. (2). An efficient solution of the NNLS subproblem Eq. (5) forms the core of FNMA^{E}. Hence, first we focus our attention on efficiently solving the NNLS problem.

Broadly viewed, our method for solving NNLS may be viewed as combining the active set method with the projected gradient scheme. This approach is founded upon the observation that if the constraints active at the final solution are known in advance, the original problem can be solved by optimizing the objective in an equality-constrained manner over only the variables that correspond to the inactive constraints.

However, by itself, the projected gradient method, being a direct analog of steepest descent, suffers from deficiencies such as slow convergence and zigzagging. For *unconstrained* optimization problems, it is known that the use of nondiagonal positive definite gradient scaling matrices alleviates such problems. To overcome problems associated with gradient-based methods, Bertsekas [16] developed a projection framework for simply constrained cases based on the Newton-method. We build on that idea and employ nondiagonal gradient scaling based on the quasi-Newton method for Problem Eq. (5), which is a *constrained* minimization problem. However, since the constraints are particularly simple, this approach remains feasible and relatively simple.

3.1. Overview of our Method for NNLS

Our algorithm for solving Eq. (5) is iterative and at each iteration it partitions the variables into two groups, namely the *free* and *fixed* variables. The fixed variables are the components of $\boldsymbol{x}^k$ with active constraints (equality satisfied) that have a corresponding positive derivative at iteration k. We index them by the *fixed set*, i.e.

$$I_+ = \{i | x_i^k = 0, [\nabla f(\boldsymbol{x}^k)]_i > 0\}. \tag{8}$$

For brevity, we will slightly manipulate the notation and say that $x_i^k \in I_+$ whenever $i \in I_+$.

We denote the free variables and the fixed variables at iteration k by $\boldsymbol{y}^k$ and $\boldsymbol{z}^k$ respectively. Without loss of generality, we can assume that $\boldsymbol{x}^k$ and $\nabla f(\boldsymbol{x}^k)$ are partitioned as

$$\boldsymbol{x}^k = \begin{bmatrix} \boldsymbol{y}^k \\ \boldsymbol{z}^k \end{bmatrix}, \quad \nabla f(\boldsymbol{x}^k) = \begin{bmatrix} \nabla f(\boldsymbol{y}^k) \\ \nabla f(\boldsymbol{z}^k) \end{bmatrix},$$

where $y_i^k \notin I_+$ and $z_i^k \in I_+$. Once the free variables at the current iteration are identified, we compute the projection $\boldsymbol{y}$ (onto the nonnegative orthant) as follows

$$\boldsymbol{y} = \mathcal{P}_+[\boldsymbol{y}^k - \alpha \overline{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)], \tag{9}$$

where $\alpha \geq 0$, and $\overline{\boldsymbol{D}}^k$ is an appropriate positive definite gradient scaling matrix. Note that $\nabla f(\boldsymbol{y}^k)$ is the gradient vector restricted to the free variables, and $\overline{\boldsymbol{D}}^k$ is a corresponding restricted scaling matrix.

Finally, given $\boldsymbol{y}$ we update $\boldsymbol{x}^k$ to obtain

$$\boldsymbol{x}^{k+1} \leftarrow \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z}^k \end{bmatrix} = \begin{bmatrix} \mathcal{P}_+[\boldsymbol{y}^k - \alpha \overline{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)] \\ \boldsymbol{0} \end{bmatrix}, \tag{10}$$

where the last equality uses the fact that $\boldsymbol{z}^k$ is fixed to zero. Now we can compute $\nabla f(\boldsymbol{x}^{k+1})$ and update the fixed set I_+ to obtain $\boldsymbol{y}^{k+1}$ and $\boldsymbol{z}^{k+1}$.

Note that any algorithm that finds $\boldsymbol{y}$ such that

$$g^k(\boldsymbol{y}) < g^k(\boldsymbol{y}^k), \quad \boldsymbol{y} \geq 0, \tag{11}$$

where

$$g^k(\boldsymbol{y}) = \tfrac{1}{2} \| \boldsymbol{G}[\boldsymbol{y}; \boldsymbol{z}^k] - \boldsymbol{h} \|_2^2, \tag{12}$$

can be used to update $\boldsymbol{x}^k$ in Eq. (10), but since Eq. (11) is again a constrained problem, Eq. (9) remains a good choice for feasibility and efficiency of the overall algorithm. Furthermore, due to the resemblance of Eq. (9) to an iteration of the standard quasi-Newton update, it is possible to exploit the curvature information of g^k to obtain a faster convergence rate.

However, the computation of a proper $\overline{\boldsymbol{D}}^k$ at each iteration is not a trivial task as the size of $\boldsymbol{y}^k$ may vary across iterations, and it may be necessary to vary the size of $\overline{\boldsymbol{D}}^k$ from one iteration to the next. To address this difficulty, we note that the curvature information from $\{\boldsymbol{y}^k\}$ is essentially captured by the sequence $\{\boldsymbol{x}^k\}$. Therefore, $\overline{\boldsymbol{D}}^k$ can be approximated by taking a proper sub-matrix of $\boldsymbol{D}^k$, which contains curvature information from the vectors $\{\boldsymbol{x}^k\}$. On the basis of the above rationale, we maintain a gradient scaling matrix $\boldsymbol{D}^k$ that covers the entire vector $\boldsymbol{x}^k$ at each iteration and build the restricted matrices $\overline{\boldsymbol{D}}^k$ from $\boldsymbol{D}^k$ according to the free variables $\boldsymbol{y}^k$.

There are many possible choices for $\boldsymbol{D}^k$, ranging from the identity matrix to the inverse of the Hessian. We provide three well-established schemes for selecting the gradient scaling matrix $\boldsymbol{D}^k$: (i) the inverse of the Hessian, (ii) the BFGS update and (iii) the memoryless BFGS update [16]. Briefly, the use of inverse Hessian, which leads to the Newton method for unconstrained problems,

is suitable when the Hessian is available and sparse.[1] The BFGS update incrementally approximates the inverse of the Hessian using only gradient information at each iteration, whereas it is recommended for problems where the computation involving the Hessian is expensive. The memoryless BFGS is for larger problems where the storage for $\boldsymbol{D}^k$ itself is too expensive. For the least squares objective function, the memoryless BFGS update becomes equivalent to the conjugate gradient method.

3.1.1. The BFGS update

Suppose $\boldsymbol{H}^k$ is the current approximation to the Hessian. The BFGS update adds a rank-two correction to $\boldsymbol{H}^k$ to obtain

$$\boldsymbol{H}^{k+1} = \boldsymbol{H}^k - \frac{\boldsymbol{H}^k \boldsymbol{u}\boldsymbol{u}^T \boldsymbol{H}^k}{\boldsymbol{u}^T \boldsymbol{H}^k \boldsymbol{u}} + \frac{\boldsymbol{w}\boldsymbol{w}^T}{\boldsymbol{u}^T \boldsymbol{w}}, \tag{13}$$

where $\boldsymbol{w} = \nabla f(\boldsymbol{x}^{k+1}) - \nabla f(\boldsymbol{x}^k)$, and $\boldsymbol{u} = \boldsymbol{x}^{k+1} - \boldsymbol{x}^k$. Let $\boldsymbol{D}^k$ denote the inverse of $\boldsymbol{H}^k$, then applying the Sherman-Morrison-Woodbury formula to Eq. (13) yields

$$\boldsymbol{D}^{k+1} = \boldsymbol{D}^k - \frac{(\boldsymbol{D}^k \boldsymbol{w}\boldsymbol{u}^T + \boldsymbol{u}\boldsymbol{w}^T \boldsymbol{D}^k)}{\boldsymbol{u}^T \boldsymbol{w}} + \left(1 + \frac{\boldsymbol{w}^T \boldsymbol{D}^k \boldsymbol{w}}{\boldsymbol{u}^T \boldsymbol{w}}\right) \frac{\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T \boldsymbol{w}}. \tag{14}$$

Since, $\nabla f(\boldsymbol{x}^k) = \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{x}^k - \boldsymbol{G}^T \boldsymbol{h}$ for the NNLS problem, Eq. (14) can be rewritten as

$$\boldsymbol{D}^{k+1} = \boldsymbol{D}^k - \frac{(\boldsymbol{D}^k \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}\boldsymbol{u}^T + \boldsymbol{u}\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{D}^k)}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}} + \left(1 + \frac{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{D}^k \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}}\right) \frac{\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}}. \tag{15}$$

REMARK 1: Note that $\|\boldsymbol{G}\boldsymbol{u}\|_2^2$ appears as the denominator in the last two terms of Eq. (15). When $\boldsymbol{G}$ is of full-rank, Eq. (15) is always well-defined, since

$$\|\boldsymbol{G}\boldsymbol{u}\|_2 = 0, \quad \text{iff} \quad \boldsymbol{u} = \boldsymbol{0},$$

which in turn implies that the method has converged and the update Eq. (15) is not needed anymore. In general, even if $\boldsymbol{G}$ is rank-deficient, we can avoid trouble by simply bypassing the update. The only requirement that we need to satisfy is that $\boldsymbol{D}^{k+1}$ remains positive and definite, which can be easily satisfied by simply setting $\boldsymbol{D}^{k+1} = \boldsymbol{D}^k$. However, in practice Eq. (15) remains well-defined, and we do not usually encounter $\|\boldsymbol{G}\boldsymbol{u}\| = 0$.

[1] In the implementation, we use the Q-less QR-decomposition of the Hessian instead of computing the inverse Hessian.

3.1.2. Memoryless BFGS update

By simply resetting the matrix $\boldsymbol{D}^k$ to an identity matrix at each iteration k, we obtain the memoryless BFGS update. Assume that at each iteration, we compute an optimal parameter α in Eq. (9). It can be shown that the equality

$$\nabla f(\boldsymbol{x}^{k+1})^T \boldsymbol{u} = 0$$

holds under this assumption. Let $\boldsymbol{d}^{k+1} = -\boldsymbol{D}^{k+1} \nabla f(\boldsymbol{x}^{k+1})$, then from Eq. (14),

$$\begin{aligned}
\boldsymbol{d}^{k+1} &= -\boldsymbol{D}^{k+1} \nabla f(\boldsymbol{x}^{k+1}) \\
&= -\nabla f(\boldsymbol{x}^{k+1}) - \left(1 + \frac{\boldsymbol{w}^T \boldsymbol{w}}{\boldsymbol{u}^T \boldsymbol{w}}\right) \frac{\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T \boldsymbol{w}} \nabla f(\boldsymbol{x}^{k+1}) \\
&\quad + \frac{(\boldsymbol{w}\boldsymbol{u}^T + \boldsymbol{u}\boldsymbol{w}^T)}{\boldsymbol{u}^T \boldsymbol{w}} \nabla f(\boldsymbol{x}^{k+1}) \\
&= -\nabla f(\boldsymbol{x}^{k+1}) + \frac{\boldsymbol{u}\boldsymbol{w}^T}{\boldsymbol{u}^T \boldsymbol{w}} \nabla f(\boldsymbol{x}^{k+1}) \\
&= -\nabla f(\boldsymbol{x}^{k+1}) + \left(\frac{\nabla f(\boldsymbol{x}^{k+1})^T \boldsymbol{w}}{\boldsymbol{u}^T \boldsymbol{w}}\right) \boldsymbol{u} \\
&= -\nabla f(\boldsymbol{x}^{k+1}) + \left(\frac{\nabla f(\boldsymbol{x}^{k+1})^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}}\right) \boldsymbol{u}.
\end{aligned}$$

3.1.3. Line-search

From Eq. (9) we see that in addition to the computation of $\boldsymbol{D}^k$, the update also involves a parameter $\alpha > 0$. Like many other iterative optimization procedures, standard line-search methods can be used to choose the step-size α. We omit a discussion of the same for brevity and refer the reader to ref. [8].

3.2. Convergence

In this section, we prove that our method for NNLS as described above is an exact method, i.e. it converges to the globally optimal solution of Eq. (5). The main result of this section is the following theorem.

THEOREM 1: (Convergence and Optimality). If $\boldsymbol{G}$ is of full-rank and $\{\boldsymbol{x}^k\}$ is the sequence of points generated by Eq. (10), then every limit point of $\{\boldsymbol{x}^k\}$ is a stationary point of Problem Eq. (5), and hence optimal since Eq. (5) is strictly convex.

The proof of this theorem depends on several lemmas that we prove below. Our proof is structured as follows. First we show that the update Eq. (10) ensures a monotonic

descent in the objective function value (Lemma 1). Then we show that the resulting sequence of iterates $\{\boldsymbol{x}^k\}$ has a limit-point (Lemma 2). Finally, we show that any limit point of the sequence $\{\boldsymbol{x}^k\}$ is also a stationary or KKT point of Eq. (5), thereby concluding the proof.

LEMMA 1: (Descent). If $\boldsymbol{x}^k$ is not a stationary point of Eq. (5), then there exists some constant $\overline{\alpha}$ such that

$$f(\boldsymbol{x}^{k+1}) < f(\boldsymbol{x}^k), \quad \forall \alpha \in (0, \overline{\alpha}],$$

where $\boldsymbol{x}^{k+1}$ and f($\boldsymbol{x}$) are given by Eq. (10) and Eq. (5) respectively.

PROOF: By the construction of I_+, all components of $\boldsymbol{y}^k$ satisfy:

$$\text{either } y_i^k \neq 0 \text{ or } [\nabla f(\boldsymbol{y}^k)]_i \leq 0.$$

Furthermore, since $\boldsymbol{x}^k$ is not a stationary point, there exists at least one i such that

$$[\nabla f(\boldsymbol{y}^k)]_i \neq 0.$$

Thus letting $\boldsymbol{d} = -\overline{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)$, we see that

$$\nabla f(\boldsymbol{y}^k)^T \boldsymbol{d} < 0,$$

since $\overline{\boldsymbol{D}}^k$ is a principal submatrix of the positive definite matrix $\boldsymbol{D}^k$, and is therefore itself positive definite. This establishes the fact that $\boldsymbol{d}$ is a feasible descent direction. Now let $\gamma(\alpha) = \boldsymbol{y}^k + \alpha \boldsymbol{d}$ denote a step in the direction given by $\boldsymbol{d}$ and consider partitioning the *free* variables into two disjoint sets of indices such that

$$I_1 = \{i \,|\, y_i^k > 0 \text{ or } (y_i^k = 0 \text{ and } d_i \geq 0)\}, \text{ and } I_2 = \{i \,|\, y_i^k = 0 \text{ and } d_i < 0\}.$$

It is easy to see that there exists $\alpha_1 > 0$ such that $\forall_i \in I_1$,

$$y_i^k + \alpha d_i \geq 0, \quad \forall \alpha \leq \alpha_1.$$

Let us define a new search direction $\overline{\boldsymbol{d}}$,

$$\overline{d}_i = \begin{cases} d_i, & i \in I_1, \\ 0, & \text{otherwise.} \end{cases}$$

Then we have,

$$\mathcal{P}_+[\gamma(\alpha)] = \boldsymbol{y}^k + \alpha \overline{\boldsymbol{d}}, \quad \forall \alpha \in (0, \alpha_1].$$

Since $[\nabla f(\boldsymbol{y}^k)]_i \leq 0$ and $d_i < 0$ for $i \in I_2$, we get $\sum_{i \in I_2} [\nabla f(\boldsymbol{y}^k)]_i \cdot d_i \geq q0$. Now we can conclude that

$$\nabla f(\boldsymbol{y}^k)^T \overline{\boldsymbol{d}} = \sum_{i \in I_1} [\nabla f(\boldsymbol{y}^k)]_i \cdot d_i \leq \sum_{i \in \{I_1 \cup I_2\}} [\nabla f(\boldsymbol{y}^k)]_i \times d_i = \nabla f(\boldsymbol{y}^k)^T \boldsymbol{d} < 0.$$

Hence, $\overline{\boldsymbol{d}}$ is also a feasible descent direction. Therefore, letting $\boldsymbol{y} = \mathcal{P}_+[\gamma(\alpha)]$, there exists $\overline{\alpha} \in (0, \alpha_1]$ such that

$$g^k(\boldsymbol{y}) < g^k(\boldsymbol{y}^k), \quad \forall \alpha \in (0, \overline{\alpha}]$$

where g^k is as in Eq. (12). From Eq. (10), since $\boldsymbol{z}^k$ remains fixed in $\boldsymbol{x}^{k+1}$, we conclude that

$$f(\boldsymbol{x}^{k+1}) < f(\boldsymbol{x}^k), \quad \forall \alpha \in (0, \overline{\alpha}].$$

LEMMA 2: (Limit point). Let $\{\boldsymbol{x}^k\}$ be a sequence of points generated by Eq. (10). Then this sequence has a limit point.

PROOF: Assume that we start the iteration at $\boldsymbol{x}^0$ where $f(\boldsymbol{x}^0) = M$. By Lemma 1, $\{f(\boldsymbol{x}^k)\}$ is a monotonically decreasing sequence, whereby $\boldsymbol{x}^0$ is a maximizer of f over the M-level set of f. If a convex quadratic function f is bounded above, its M-level set is also bounded. Denote this M-level set by $\mathbb{X}$. Then we can choose $\boldsymbol{u} \in \mathbb{X}$ such that

$$\|\boldsymbol{u}\|_2 \geq \|\boldsymbol{x}\|_2, \forall \boldsymbol{x} \in \mathbb{X}.$$

Then $\{\boldsymbol{x}^k\}$ is bounded as $0 \leq \|\boldsymbol{x}^k\|_2 \leq \|\boldsymbol{u}\|_2$ for all k, hence the sequence has a limit point. This concludes the proof of the lemma.

3.2.1. *Gradient related condition*

Let $\{\boldsymbol{x}^k\}$ be a sequence generated by Eq. (10). Then for any subsequence $\{\boldsymbol{x}^k\}_{k \in \mathcal{K}}$ that converges to a nonstationary point,

$$\limsup_{t \to \infty} \|\boldsymbol{x}^{k_{t+1}} - \boldsymbol{x}^{k_t}\| < \infty, \tag{16}$$

$$\limsup_{t \to \infty} \nabla f(\boldsymbol{x}^{k_t})^T (\boldsymbol{x}^{k_{t+1}} - \boldsymbol{x}^{k_t}) < 0. \tag{17}$$

This is known as the *gradient related condition* in optimization literature and plays a crucial role to prove the convergence of a number of methods. Inequality Eq. 16 follows from Lemma 2 and it can be shown that our method also satisfies condition Eq. (17) [8].

Finally we present a proof of our main Theorem 1.

PROOF: Assume $\{\boldsymbol{x}^k\}$ converges to a nonstationary point $\overline{\boldsymbol{x}}$. From Lemma 1, it can be shown that there exists some $\in_k$ such that $\lim_{k\to\infty} \in_k = 0$ and

$$f(\boldsymbol{x}^k) - f(\boldsymbol{x}^{k+1}) = -\nabla f(\boldsymbol{x}^k)^T(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k) - \in_k > 0,$$

Since f is continuous, $\lim_{k\to\infty} f(\boldsymbol{x}^k) = f(\overline{\boldsymbol{x}})$. Consequently,

$$\lim_{k\to\infty} f(\boldsymbol{x}^k) - f(\boldsymbol{x}^{k+1}) = 0.$$

In turn, it implies

$$\lim_{k\to\infty} \nabla f(\boldsymbol{x}^k)^T(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k) = 0,$$

which contradicts Eq. (17).

3.3. FNMA$^{\mathrm{E}}$: An *Exact* Method for NNMA

Now we extend the ideas from Section 3.1 to the matrix case. To that end, we need to redefine various quantities in terms of matrices. First, observe that the gradient matrices $\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C})$ and $\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C})$ are

$$\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = \boldsymbol{B}^T\boldsymbol{B}\boldsymbol{C} - \boldsymbol{B}^T\boldsymbol{A}, \quad \text{and}$$
$$\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = \boldsymbol{B}\boldsymbol{C}\boldsymbol{C}^T - \boldsymbol{A}\boldsymbol{C}^T.$$

Then we redefine the *fixed set* accordingly. For example, the fixed set corresponding to $\boldsymbol{B}$ is defined as:

$$I_+ = \{(i,j) | B_{ij} = 0,\ [\nabla_B\mathcal{F}(\boldsymbol{B};\boldsymbol{C})]_{ij} > 0\}.$$

Finally, we define the *zero-out operator* $\mathcal{Z}_+$ with respect to the fixed set I_+ so that

$$[\mathcal{Z}_+[\boldsymbol{X}]]_{ij} = \begin{cases} X_{ij}, & (i,j) \notin I_+, \\ 0, & \text{otherwise.} \end{cases} \tag{18}$$

Algorithm 1 FNMA$^{\mathrm{E}}$

Input: $\boldsymbol{A} \in \mathbb{R}_+^{M\times N}$, K s.t. $1 \le K \le \min\{M,N\}$
Output: $\boldsymbol{B} \in \mathbb{R}_+^{M\times K}, \boldsymbol{C} \in \mathbb{R}_+^{K\times N}$
1. Initialize $\boldsymbol{B}^0, \boldsymbol{C}^0, t = 0, \boldsymbol{S} = \boldsymbol{I}$.
repeat
  2. $\boldsymbol{B} \leftarrow \boldsymbol{B}^t$, $\boldsymbol{C}^{\mathrm{old}} \leftarrow \boldsymbol{C}^t$.
  repeat
    3.1. Compute the gradient matrix $\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}^{\mathrm{old}})$.
    3.2. Compute fixed set I_+ for $\boldsymbol{C}^{\mathrm{old}}$.
    3.3. Compute the step length vector α using line-search.
    3.4. Update $\boldsymbol{C}^{\mathrm{old}}$ as

$$\boldsymbol{U} \leftarrow \mathcal{Z}_+[\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}^{\mathrm{old}})]; \quad \boldsymbol{U} \leftarrow \mathcal{Z}_+[\boldsymbol{S}\boldsymbol{U}];$$
$$\boldsymbol{C}^{\mathrm{new}} \leftarrow \mathcal{P}[\boldsymbol{C}^{\mathrm{old}} - \boldsymbol{U}\cdot\mathrm{diag}(\alpha)].$$

    3.5. $\boldsymbol{C}^{\mathrm{old}} \leftarrow \boldsymbol{C}^{\mathrm{new}}$.
    3.6. Update $\boldsymbol{S}$ if necessary.
  until $\boldsymbol{C}^{\mathrm{old}}$ converges
  4. $\boldsymbol{C}^{t+1} \leftarrow \boldsymbol{C}^{\mathrm{old}}$.
  5. Repeat steps similar to Step 2–4 to obtain $\boldsymbol{B}^{t+1}$.
  6. $t \leftarrow t+1$.
until Stopping criteria are met

Now we have all the pieces to describe the overall algorithm for solving the NNMA problem Eq. (2). Algorithm 1 presents our proposed method which we name fast nonnegative matrix approximationly—exact, i.e. FNMA$^{\mathrm{E}}$.

In Step 3.4 of Algorithm 1, the first $\mathcal{Z}_+[\cdot]$ eliminates the "fixed" gradient information from the search direction, the second $\mathcal{Z}_+[\cdot]$ ensures that the fixed set remains fixed, and the projection $\mathcal{P}_+[\cdot]$ maintains feasibility of the next iterate.

Note that we maintain only one gradient scaling matrix $\boldsymbol{D}$ at each alternating step even though our algorithm for NNLS suggests that each column should have its own gradient scaling matrix. We justify this strategy as follows. In Problem Eq. (5), a series of BFGS updates for $\boldsymbol{D}$ aim at estimating the inverse of the Hessian. However, the true Hessian for Problem Eq. (5) is $\boldsymbol{G}^T\boldsymbol{G}$ which is a constant matrix. Thus in Problem Eq. (2), a matrix-wise extension of NNLS, every column shares the same true Hessian, whereby each column can also share the approximation of the inverse Hessian, namely $\boldsymbol{D}$. As long as we retain the positive definiteness of the matrix $\boldsymbol{D}$, this shared $\boldsymbol{D}$ provides an effective gradient scaling, and it does not impede convergence of the algorithm. Also note that since the Hessian is of size $K \times K$, its exact inverse can be used if K is not too large at a computational cost of $O(K^2(M+N) + K^3)$ operations. This strategy is included in our second algorithm, FNMA$^{\mathrm{I}}$, in the next section.

THEOREM 2: (Convergence of FNMA$^{\mathrm{E}}$). If $\boldsymbol{B}^t$ and $\boldsymbol{C}^t$ retain full-rank, then the sequence $\{\boldsymbol{B}^t, \boldsymbol{C}^t\}$ generated by Algorithm $FNMA^E$ converges to a stationary point of Problem Eq. (2).

PROOF: Algorithm 1 essentially performs the following alternating minimization at each outer iteration

$$\boldsymbol{C}^{t+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{C}\ge 0} \|\boldsymbol{A} - \boldsymbol{B}^t\boldsymbol{C}\|_{\mathrm{F}}^2, \text{ and } \boldsymbol{B}^{t+1}$$
$$\leftarrow \operatorname*{argmin}_{\boldsymbol{B}\ge 0} \|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}^{t+1}\|_{\mathrm{F}}^2.$$

Similar to the argument in Lemma 2, the domain of Problem Eq. (2) can be considered to be compact. Since $\{\mathcal{F}(\boldsymbol{B}^t; \boldsymbol{C}^t)\}$ is monotone decreasing and bounded below, it has a limit point, $(\overline{\boldsymbol{B}}, \overline{\boldsymbol{C}})$, i.e.

$$\lim_{t\to\infty} \mathcal{F}(\boldsymbol{B}^t; \boldsymbol{C}^t) = \mathcal{F}(\overline{\boldsymbol{B}}; \overline{\boldsymbol{C}}).$$

Since $\mathcal{F}$ is continuous, we have

$$\lim_{t\to\infty} \boldsymbol{B}^t = \overline{\boldsymbol{B}}, \text{ and } \lim_{t\to\infty} \boldsymbol{C}^t = \overline{\boldsymbol{C}}.$$

Now we can invoke the proof of the two-block Gauss-Seidel method [17] to conclude our claim.

3.4. FNMA$^{\mathrm{I}}$: An *Inexact* Method for NNMA

In this section we present an *inexact* version of our approach. This method has the same underlying framework as FNMA$^{\mathrm{E}}$, but uses some heuristics to reduce computational effort at each iteration.

Algorithm 2 gives the pseudocode for FNMA$^{\mathrm{I}}$ and it differs from the exact method in three main aspects. First, it uses the inverse of the Hessian as the nondiagonal gradient scaling matrix $\boldsymbol{D}$. Whenever the rank K of the factor matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ is small, using the inverse Hessian can be advantageous for problems where $\boldsymbol{O}$ (K^3) costs are acceptable. Second, the step-size α is made an input parameter, and FNMA$^{\mathrm{I}}$ guarantees monotonic descent on the objective function for a sufficiently small α. Third, FNMA$^{\mathrm{I}}$ accepts the number of iterations for each alternating step as an input parameter. This modification permits premature termination of each alternating step, which naturally translates into large computational savings by trading-off accuracy for speed.

THEOREM 3: (Monotonicity of FNMA$^{\mathrm{I}}$). If $\boldsymbol{B}^t$ and $\boldsymbol{C}^t$ retain full-rank, then $FNMA^I$ decreases its objective function monotonically for sufficiently small α.

PROOF: It is enough to consider Steps 2–3 from FNMA$^{\mathrm{I}}$ (the argument for Steps 4–5 is similar). Since $\boldsymbol{B}$ is assumed to be full-rank at every step, $(\boldsymbol{B}^T\boldsymbol{B})^{-1}$ is positive definite. For a sufficiently small α that satisfies

$$\alpha \le \min\{\alpha_i, i = 1, \cdots, K\},$$

Algorithm 2 FNMA$^{\mathrm{I}}$

Input: $\boldsymbol{A} \in \mathbb{R}_+^{M\times N}$, $K, \tau \in \mathbb{N}, \alpha \in \mathbb{R}_+$.
Output: $\boldsymbol{B} \in \mathbb{R}_+^{M\times K}$, $\boldsymbol{C} \in \mathbb{R}_+^{K\times N}$
1. Initialize $\boldsymbol{B}^0, \boldsymbol{C}^0, t = 0$.
repeat
2. $\boldsymbol{B} \leftarrow \boldsymbol{B}^t$, $\boldsymbol{C}^{\text{old}} \leftarrow \boldsymbol{C}^t$.
for $i = 1$ to τ **do**
3.1. Compute the gradient matrix $\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B}; \boldsymbol{C}^{\text{old}})$.
3.2. Compute fixed set I_+ for $\boldsymbol{C}^{\text{old}}$.
3.3. Update $\boldsymbol{C}^{\text{old}}$ as:

$$\boldsymbol{U} \leftarrow \mathcal{Z}_+[\nabla_C\mathcal{F}(\boldsymbol{B}; \boldsymbol{C}^{\text{old}})]; \quad \boldsymbol{U} \leftarrow \mathcal{Z}_+[(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{U}];$$
$$\boldsymbol{C}^{\text{new}} \leftarrow \mathcal{P}[\boldsymbol{C}^{\text{old}} - \alpha\boldsymbol{U}].$$

3.4. $\boldsymbol{C}^{\text{old}} \leftarrow \boldsymbol{C}^{\text{new}}$.
end for
4. $\boldsymbol{C}^{t+1} \leftarrow \boldsymbol{C}^{\text{old}}$.
5. Repeat steps similar to Step 2–4 to obtain $\boldsymbol{B}^{t+1}$.
6. $t \leftarrow t + 1$.
until Stopping criteria are met

where the α_i are computed by Step 3.3 from FNMA$^{\mathrm{E}}$, it can be shown that Steps 2-3 decrease the objective monotonically by arguments similar to the ones in the proof of Lemma 1.

REMARK 2: If any α_i is zero, then the inner loop for the current alternating step should be terminated to guarantee monotonicity.

REMARK 3: A sufficiently small α is important to guarantee monotonicity of FNMA$^{\mathrm{I}}$, but too small a value will hurt the computational benefit by slowing down convergence. On the other hand, if α is too large, it can push the search direction out of the feasible region, or introduce too many zeros into the current iterate, resulting in a singular or ill-conditioned Hessian for the next iterate.

To overcome these subtleties and to find a proper α in practice, the following simple heuristic can be used. Writing Step 3.3 from FNMA$^{\mathrm{I}}$ in the form:

$$\boldsymbol{W} \leftarrow \mathcal{P}_+[\boldsymbol{W} - \alpha\boldsymbol{U}],$$

1. Let number of *inner* iterations be small ($\tau = 2$ or 3),
2. Start with a large scaling λ (typically 0.1) and compute

$$\alpha = \lambda\frac{\|\boldsymbol{W}\|_F}{\|\boldsymbol{U}\|_F},$$

for each alternating step,
3. Decrease λ until it passes the inner steps without error,
4. Increase the number of iterations (typically $\tau = 10$).

3.5. Extensions to Handle Regularization

The regularized NNMA problem Eq. (3) can be solved by suitably modifying the FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ procedures. Essentially the gradient and Hessian get redefined. For example, the gradient

$$\nabla_{\boldsymbol{C}} \mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = (\boldsymbol{B}^T\boldsymbol{B} + \lambda \boldsymbol{I})\boldsymbol{C} - \boldsymbol{B}^T\boldsymbol{A},$$

and the Hessian

$$\nabla^2_{\boldsymbol{C}} \mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = (\boldsymbol{B}^T\boldsymbol{B} + \lambda \boldsymbol{I}),$$

are suitably modified to include the contribution of the regularization term. We just use these updated values in the algorithms FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ to handle regularization. Notice that regularization provides the benefit of ensuring that the Hessian remains positive-definite. All the convergence results carry over without any additional work.

3.6. Handling Box-Constraints

FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ can be easily extended to handle box-constraints, i.e. constraints of the form $\boldsymbol{p} \le \boldsymbol{x} \le \boldsymbol{q}$. We motivate the details by first looking at the box-constrained version of Eq. (5), which is also known as *bounded least squares* (BLS) (Björck, 1996),

$$\begin{array}{ll} \underset{\boldsymbol{x}}{\text{minimize}} & \frac{1}{2}\|\boldsymbol{G}\boldsymbol{x} - \boldsymbol{h}\|^2, \\ \text{subject to} & \boldsymbol{p} \le \boldsymbol{x} \le \boldsymbol{q}. \end{array} \tag{19}$$

Problem Eq. (19) can be solved just as we solved Eq. (2.1). We need to modify the definition of the *fixed-set* Eq. (8) so that

$$I_+ = \{i \,|\, (x_i^k = p_i, [\nabla f(\boldsymbol{x}^k)]_i > 0) \text{ or } (x_i^k = q_i, [\nabla f(\boldsymbol{x}^k)]_i < 0)\},$$

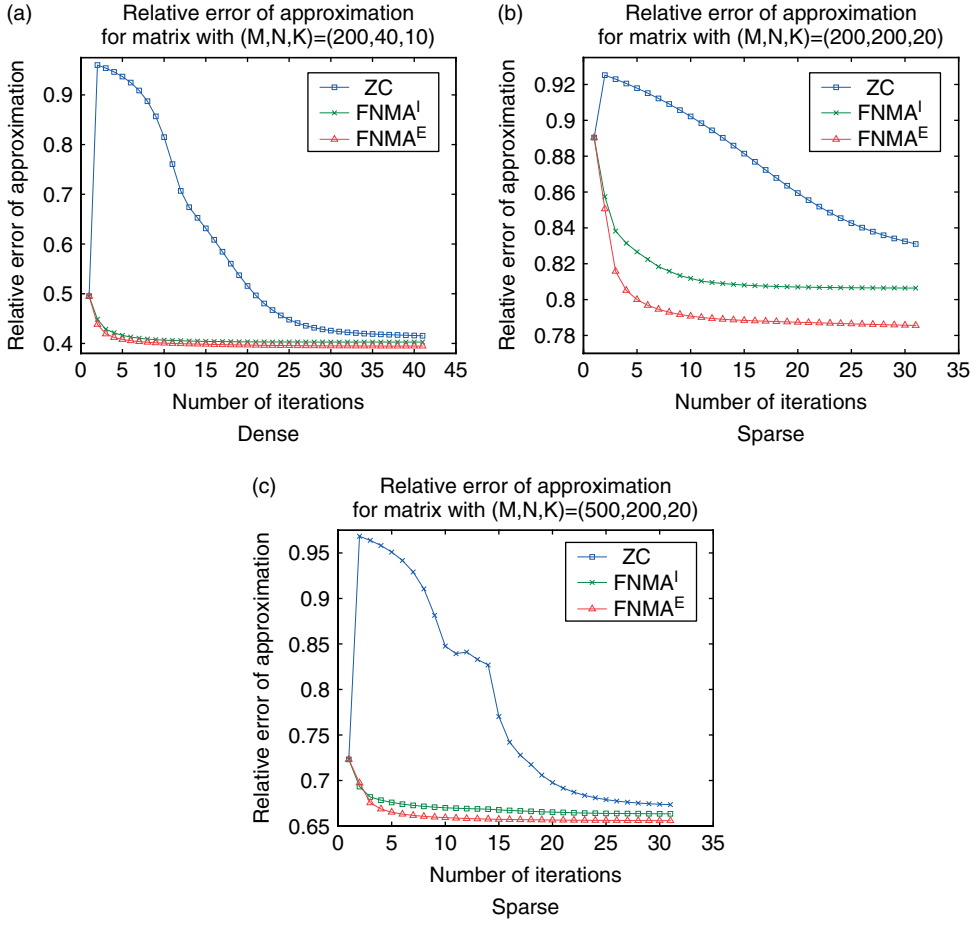

Fig. 2 Relative error of approximation against iteration count for ZC, FNMA$^{\mathrm{I}}$, and FNMA$^{\mathrm{E}}$. The relative errors achieved by both FNMA$^{\mathrm{I}}$ and FNMA$^{\mathrm{E}}$ are lower than ZC. Note that ZC does not decrease the errors monotonically.

and to replace the $\mathcal{P}_+[\cdot]$ projection by $\mathcal{P}_\Omega[\cdot]$, where

$$[\mathcal{P}_\Omega[\boldsymbol{x}]]_i = \begin{cases} p_i & : \ x_i \leq p_i \\ x_i & : \ p_i < x_i < q_i \\ q_i & : \ q_i \leq x_i \end{cases} \tag{20}$$

Given these definitions, it can be verified that Lemma 1 holds without significant modification and Theorem 1 also follows. The fact that the domain of Eq. (19) is a compact set obviates the need for Lemma 2 in this case.

Given the above method for BLS we can appropriately modify FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ for solving the bounded matrix approximation (BMA) problem Eq. (4). We omit the details for brevity, noting that the modifications needed are minor, for example, the fixed set for $\boldsymbol{B}$ is redefined as

$$I_\Omega = \{(i,j) | (B_{ij} = P_{ij}, [\nabla_{\boldsymbol{B}} \mathcal{F}(\boldsymbol{B};\boldsymbol{C})]_{ij} > 0), \text{ or } (B_{ij} = Q_{ij}, [\nabla_{\boldsymbol{B}} \mathcal{F}(\boldsymbol{B};\boldsymbol{C})]_{ij} < 0)\}.$$

By taking a projection step similar to Eq. (20) we can construct the desired method.

4. EXPERIMENTS

We now present experimental results to demonstrate the performance of our FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ methods. We give numerical results to assess the performance of our methods as compared to the standard LS method [4], ZC method [15], and the ALS approach [1] for solving the least-squares NNMA problem. Our experiments show that FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ produce better quality approximations than the LS, ZC, and the ALS procedures. We implemented LS, ALS, FNMA$^{\mathrm{E}}$, and FNMA$^{\mathrm{I}}$ in MATLAB, while the ZC method was available in the NMFLAB toolbox [18]. We present results for the ZC method only with small matrices as the implementation available in NMFLAB was unable to run on larger matrices.

Since NNMA enjoys a vast number of applications [19], all of them stand to benefit from our new methods, especially because our methods achieve better objective function values and come with theoretical guarantees. As an illustration, we include some simple results on image processing in Section 4.2.

4.1. Error of Approximation

For our experiments, we initialize all the methods randomly or with one step of LS. Our results below show plots of the relative error of approximation, i.e. $\|\boldsymbol{A} - \boldsymbol{BC}\|_{\mathrm{F}}/\|\boldsymbol{A}\|_{\mathrm{F}}$ against the number of iterations. However, a word of caution is in order—iterations of these different methods are not strictly comparable to each other, since some methods do more work than others in one iteration. A more interesting plot would have been "time" on the X-axis; however, at present we are unable to conduct such experiments since different implementations of each of the methods can change the running time substantially, for example, implementations that use BLAS3 versus those that do not. To perform timing comparisons, we intend to compare C/C++ implementations of these methods in the future.

4.1.1. *Comparisons against ZC*

The first experiment compares FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ against ZC on three data matrices. The results are reported in Fig. 2. As previously noted, the data matrices used are fairly small since ZC (NMFLAB) seems to be unable to cope with larger matrices. We initialized all methods using one iteration of LS, which itself was initialized randomly. However, in the figures we do not report the relative error for the random initialization as it is too large to display properly. Figure 2(a) indicates that our methods outperform ZC. The differences between the three algorithms are sharper in Fig. 2(b). Also note that ZC actually increases the approximation errors after the first iteration.

4.1.2. *Comparisons against LS and ALS*

On a larger matrix, Fig. 3 shows a comparison of the approximation errors for LS, ALS, FNMA$^{\mathrm{E}}$, and FNMA$^{\mathrm{I}}$.

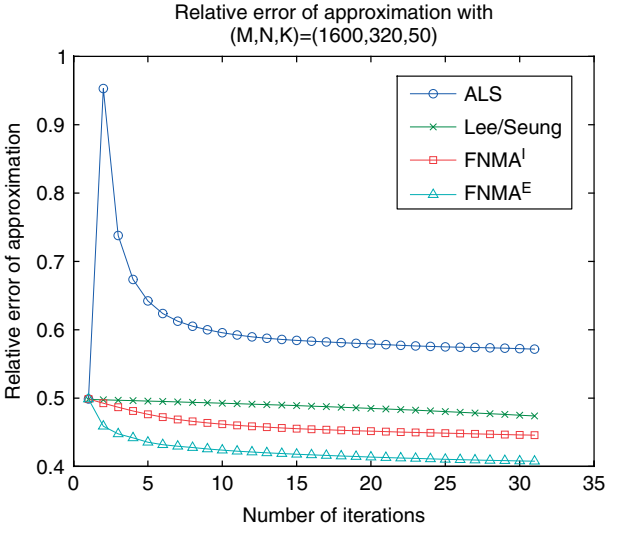

Fig. 3 Relative error values against iteration count for a random dense matrix of size 1600×320 for a rank 50 approximation. All methods other than ALS show a monotonic decrease when initialized with one step of LS.

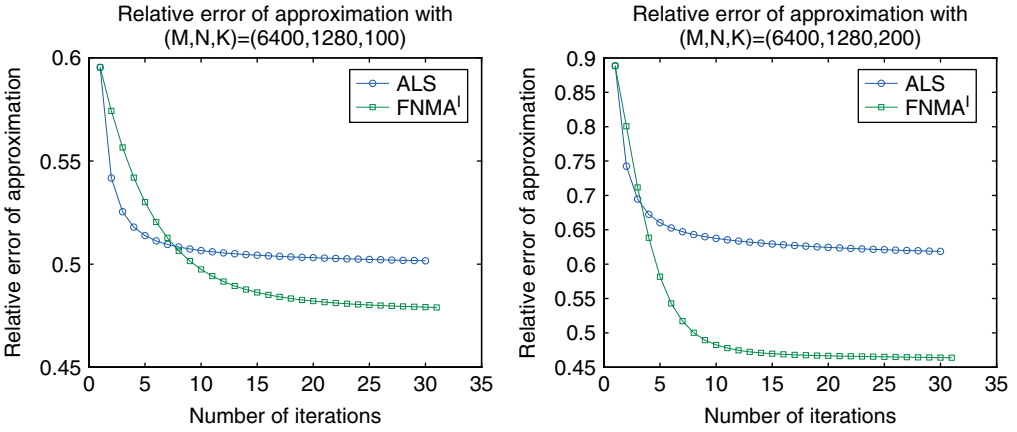

Fig. 4 Relative error values against iteration count for a random dense matrix of size 6400×1280 for a rank 100 (top) and rank 200 approximation (bottom).

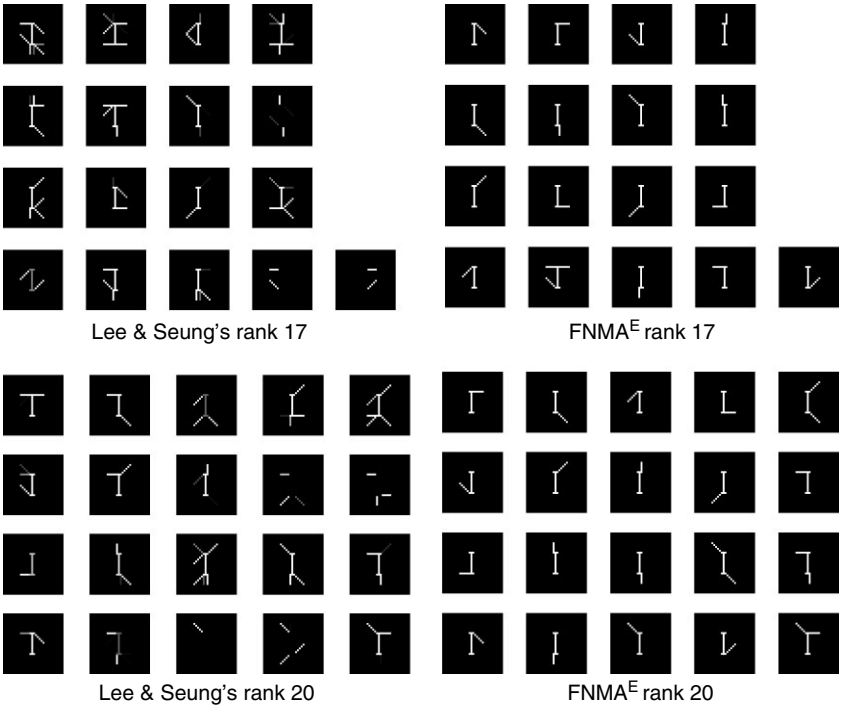

Fig. 5 Recovered nonnegative factors (matrix $\boldsymbol{B}$) from the swimmer dataset. The panels on the left are generated by Lee and Seung's algorithm while ones on the right by FNMA$^{\mathrm{E}}$. From top to bottom, the approximation is performed for rank 17 and 20, respectively.

We see that FNMA$^{\mathrm{E}}$ achieves the best objective function values of all the methods presented. However, FNMA$^{\mathrm{E}}$ can take more running time than the other methods because of its *exact* nature. Therefore, FNMA$^{\mathrm{E}}$ is to be preferred when reconstruction accuracy is more important, while FNMA$^{\mathrm{I}}$ is recommended when running time is more important. We now present two more experiments to highlight the advantages of FNMA$^{\mathrm{I}}$ over ALS, which owing to its *ad-hoc* nature leads to inferior accuracies (see also Section 2.1).

Figure 4 compares the relative errors of approximation achieved by ALS and FNMA$^{\mathrm{I}}$ for a dense random matrix of size 6400×1280. We emphasize again that the number of iterations is merely used as an indicator of progress of the algorithms, and is not to be taken as an indicator of time. From these figures one sees the interesting trend that as the rank of approximation increases, ALS becomes less and less competitive in terms of the objective function value achieved. For a rank-200 approximation (see Fig. 4),

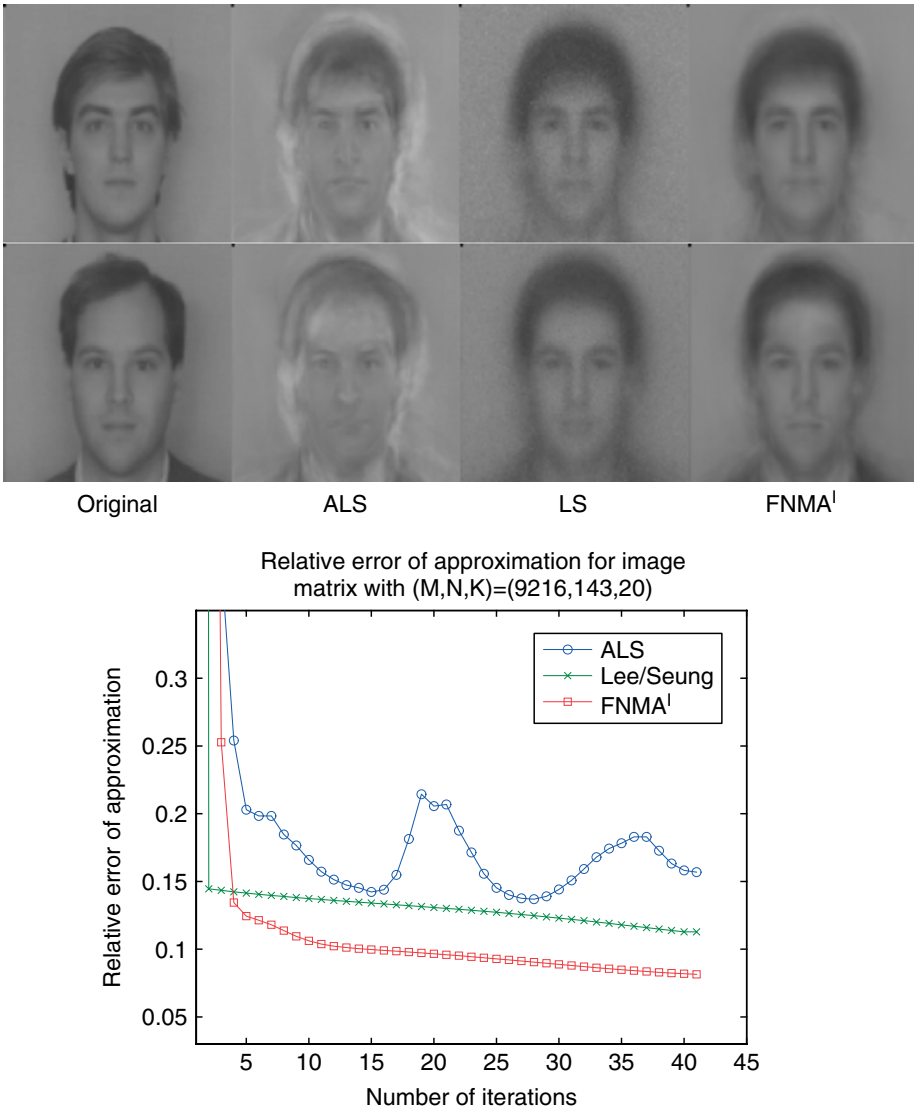

Fig. 6 Image reconstruction as obtained by the ALS, LS, and FNMA^{I} procedures. The figure illustrates two randomly chosen images out of the 143 reconstructed images, each with 96×96 pixels. The reconstruction was computed from a rank-20 approximation to the input image matrix, which was of size 9216×143. The first image in each row is the original, followed by reconstructions obtained via ALS, LS, and FNMA^{I}. From the images above, FNMA^{I} is seen to obtain the best reconstruction and the relative errors as plotted on the right attest to this observation. Observe how ALS leads to a nonmonotonic change in the objective function value (as explained in Section 2.1). Note that the error values for the initialization are not fully shown as they are too large to display properly without obscuring the rest of the plot.

the accuracy achieved by FNMA^{I} is 25% higher than that achieved by ALS.

4.2. Application to Image Processing

NNMA was originally motivated by Lee and Seung [2] using an image processing application. Many other authors have also considered NNMA for image processing, graphics, or face recognition applications. Figure 5 compares LS method to FNMA^{E} for the swimmer dataset [20]. In the figure, the effect of differences in final objective function values is more apparent. In our experiment, we run LS method up to 3000 iterations and FNMA^{E} up to 20 iterations. We also set the minimal threshold in the objective function value to be 10^{-5}; hence both methods either stop after the maximum number of iterations or if progress is below the threshold. From Table 1, we can see that FNMA^{E} generally outperforms LS method, both in terms of the elapsed CPU time and the final objective function value achieved.

Table 1. Results on the swimmer dataset.

	Lee and Seung's	FNMA^{E}	
Rank 17	182.24	62.29	Elapsed CPU time
	2.41×10^7	$\mathbf{6.85 \times 10^{-4}}$	Objective function value
Rank 20	156.18	41.93	Elapsed CPU time
	5.61×10^5	4.71×10^3	Objective function value

FNMA^{E} also produces sparser images (factors)—each image contains less number of component (limb) images. The difference becomes even more pronounced as the approximated rank approaches 17.[2]

Since, the quality of the reconstruction achieved by NNMA is important to many image processing applications, we provide a comparison of the various NNMA methods in terms of reconstruction accuracy—sample results are reported in Fig. 6, which shows accuracies for a rank-20 approximation to a 9216×143 matrix of face images.[3] All methods were initialized with the same random $\boldsymbol{B}$ and $\boldsymbol{C}$ values.

This image dataset is an example of a real-world dense matrix for which ALS fails to decrease the objective function monotonically, resulting in a corresponding poorer reconstruction accuracy. FNMA^{I} achieves the best objective values of all three algorithms compared, and a corresponding better reconstruction is observed (see Fig. 6).

5. CONCLUSIONS

In this paper, we have presented new and improved Newton-type methods for the least-squares NNMA problem. By employing a nondiagonal gradient scaling scheme, our algorithms use curvature information and thus overcome deficiencies of gradient descent-based methods. Our methods also rectify serious draw-backs in existing methods such as ALS and ZC quasi-Newton heuristic. We provide convergence guarantees for our algorithms and verify their performance on real-life data from applications.

We provide two implementations based on the same algorithmic framework. Our *exact* method FNMA^{E}, which shows good performance in terms of approximation accuracy, is suitable for applications that require superior accuracy. Our inexact implementation FNMA^{I} is more suitable for applications that are more constrained by computational efficiency rather than accuracy.

[2] In ref. [20], Donoho presents the result of a rank 16 estimation. Since we are not aware of an efficient algorithm to compute the nonnegative rank of matrix, we are not able to confirm the true nonnegative rank of the dataset. However, it obviously lies between the matrix rank 13 and the number of component images 17. We conjecture that the nonnegative rank is either 16 or 17 and our experiment confirms this conjecture by achieving almost zero error for a rank 17 approximation.

[3] We preprocessed a publicly available face image database to create a subset of 143 grey-scale images of dimension 96×96 for our experiments.

ACKNOWLEDGEMENTS

This research was supported by NSF grant CCF-0431257, NSF Career Award ACI-0093404, and NSF-ITR award IIS-0325116.

REFERENCES

[1] M. Berry, M. Browne, A. Langville, P. Pauca, and R. J. Plemmons, Algorithms and applications for approximation nonnegative matrix factorization, Comput Stat Data Anal (2006), Preprint.

[2] D. D. Lee and H. S. Seung, Learning the parts of objects by nonnegative matrix factorization, Nature 401 (1999), 788–791.

[3] P. Paatero and U. Tapper, Positive matrix factorization: A nonnegative factor model with optimal utilization of error estimates of data values, Environmetrics 5 (1994), 111–126.

[4] D. D. Lee and H. S. Seung, Algorithms for Nonnegative Matrix Factorization. In *Neural Information Processing Systems*, 2000, 556–562.

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

[6] R. Bro and S. D. Jong, A Fast Non-negativity-constrained Least Squares Algorithm, J Chemomet 11(5) (1997), 393–401.

[7] Åke Björck, Numerical Methods for Least Squares Problems, SIAM, 1996.

[8] D. Kim, S. Sra, and I. S. Dhillon, A New Projected Quasi-Newton Approach for the Non-negative Least Squares Problem. Technical Report TR-06-54, Computer Sciences, The University of Texas at Austin, 2006.

[9] P. Paatero, Least-squares formulation of robust nonnegative factor analysis, Chemomet Intell Lab Syst 37 (1997), 23–35.

[10] P. Paatero, The multilinear engine–a table-driven least squares program for solving multilinear problems, including the N-way parallel factor analysis model, J Comput Graphical Statist 8(4) (1999), 854–888.

[11] M. Bierlaire, P. L. Toint, and D. Tuyttens, On iterative algorithms for linear least squares problems with bound constraints, Linear Algebra Appl 143 (1991), 111–143.

[12] C. Lin, Projected Gradient Methods for Non-negative Matrix Factorization. Technical Report ISSTECH-95-013, National Taiwan University, 2005.

[13] M. Merritt and Y. Zhang, Interior-point gradient method for large-scale totally nonnegative least squares problems, J Optim Theory Appl 126(1) (2005), 191–202.

[14] E. F. Gonzalez and Y. Zhang, Accelerating The Lee-Seung Algorithm for Nonnegative Matrix Factorization. Technical Report TR-05-02, Rice University, 2005.
[15] R. Zdunek and A. Cichocki, Non-Negative Matrix Factorization with Quasi-Newton Optimization. In *Eighth International Conference on Artificial Intelligence and Soft Computing, ICAISC*, 2006, 870–879.
[16] D. P. Bertsekas, Projected Newton methods for optimization problems with simple constraints, SIAM J Control Optimizat 20(2) (1982), 221–246.
[17] L. Grippo and M. Sciandrone, On the convergence of the block nonlinear gauss-seidel method under convex constraints, Operat Res Lett 26 (2000), 127–136.
[18] A. Cichocki and R. Zdunek, NMFLAB–MATLAB Toolbox for Non-Negative Matrix Factorization, Online, 2006.
[19] S. Sra and I. S. Dhillon, Nonnegative Matrix Approximation: Algorithms and Applications. Technical Report Tr-06-27, Computer Sciences, University of Texas at Austin, 2006.
[20] D. Donoho and V. Stodden, When Does Nonnegative Matrix Factorization Give a Correct Decomposition into Parts? In *Neural Information Processing Systems*, 2003.