Fast Newton-type Methods for the Least Squares Nonnegative Matrix Approximation Problem

Dongmin Kim, Suvrit Sra and Inderjit S. Dhillon
Department of Computer Sciences, University of Texas
Austin, TX 78712-1188, USA
{dmkim,suvrit,inderjit}@cs.utexas.edu

Abstract

Nonnegative Matrix Approximation is an effective matrix decomposition technique that has proven to be useful for a wide variety of applications ranging from document analysis and image processing to bioinformatics. There exist a few algorithms for nonnegative matrix approximation (NNMA), for example, Lee & Seung's multiplicative updates, alternating least squares, and certain gradient descent based procedures. All of these procedures suffer from either slow convergence, numerical instabilities, or at worst, theoretical unsoundness. In this paper we present new and improved algorithms for the least-squares NNMA problem, which are not only theoretically well-founded, but also overcome many of the deficiencies of other methods. In particular, we use non-diagonal gradient scaling to obtain rapid convergence. Our methods provide numerical results superior to both Lee & Seung's method as well to the alternating least squares (ALS) heuristic, which is known to work well in some situations but has no theoretical guarantees (Berry et al. 2006). 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 to demonstrate the superiority of our methods, in terms of better approximations as well as efficiency.

Keywords: Nonnegative matrix approximation, factorization, projected Newton methods, active sets, least-squares

1 Introduction

Nonnegative matrix approximation, also known as *nonnegative matrix factorization* [15] or *positive matrix factorization* [20], is a popular and effective matrix decomposition technique. By now it has become an established method for performing dimensionality reduction and related tasks such as clustering, image processing, and visualization, to name a few. The nonnegative matrix approximation (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 $\boldsymbol{a}_i \in \mathbb{R}_+^M$. NNMA seeks to approximate these input vectors by nonnegative linear, i.e., conic, combinations of a small number of *non-negative representative vectors* $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_K$, so that

$$\boldsymbol{a}_i \approx \sum_{k=1}^{K} \boldsymbol{b}_k c_{ki}, \tag{1.1}$$

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 (1.1) 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 *least-squares NNMA* problem,

$$\underset{\boldsymbol{B},\boldsymbol{C}\geq 0}{\text{minimize}} \quad \mathcal{F}(\boldsymbol{B};\boldsymbol{C}) = \tfrac{1}{2}\|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}^2, \tag{1.2}$$

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

In this paper we develop two new Newton-type algorithms for solving (1.2) along with a theoretical analysis to establish their convergence. Both of our algorithms improve upon the *de facto* procedure of Lee & Seung [14], hereafter referred to as LS, as well as upon the popular alternating least-squares (ALS) heuristic, which has been reported to perform well in practice [1]. However, LS suffers from slow convergence and ALS lacks theoretical guarantees on its performance—our new algorithms rectify both of these deficiencies.

Researchers have also considered the following regularized NNMA problem

$$\underset{\boldsymbol{B},\boldsymbol{C}\geq 0}{\text{minimize}} \quad \tfrac{1}{2}\|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}^2 + \lambda\|\boldsymbol{B}\|_{\mathrm{F}}^2 + \mu\|\boldsymbol{C}\|_{\mathrm{F}}^2, \tag{1.3}$$

where $\lambda > 0$, and $\mu > 0$ are regularization parameters. The motivation behind studying (1.3) 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 over-fitting, which despite the apparent sparse representations yielded by NNMA, might be difficult to counter without regularization. Yet another interesting variation arises when one bounds the solution values by imposing box-constraints on the variables. For NNMA this results in the problem

$$\begin{aligned} &\text{minimize} && \tfrac{1}{2}\|\boldsymbol{A}-\boldsymbol{BC}\|_{\mathrm{F}}^2, \\ &\text{subject to} && \boldsymbol{P} \le \boldsymbol{B} \le \boldsymbol{Q}, \\ &&& \boldsymbol{R} \le \boldsymbol{C} \le \boldsymbol{S}, \end{aligned} \tag{1.4}$$

where the inequalities are component-wise. Both these variants can be handled with equal ease by our methods.

2 Background and Related Work

The NNMA objective function (1.2) is not simultaneously convex in both $\boldsymbol{B}$ and $\boldsymbol{C}$ due to the presence of the product $\boldsymbol{BC}$. Hence, in general it is very difficult to find globally optimal solutions to (1.2). However, the objective function is individually convex in $\boldsymbol{B}$ and in $\boldsymbol{C}$. Therefore, most algorithms for solving (1.2) are iterative and perform an alternating minimization or descent 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
$$\mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C}^{t+1}) \le \mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C}^t),$$
3. Fix $\boldsymbol{C}^{t+1}$ and find $\boldsymbol{B}^{t+1}$ such that
$$\mathcal{F}(\boldsymbol{B}^{t+1}, \boldsymbol{C}^{t+1}) \le \mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C}^{t+1}),$$
4. Let $t \leftarrow t+1$, and repeat Steps 2 and 3 until some convergence criteria are satisfied.

Based on the above procedure we can categorize NNMA methods into two groups, namely the *exact* and *inexact* methods. The former perform an exact minimization at each iterative step so that $\boldsymbol{C}^{t+1} = \text{argmin}_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C})$ (similarly for $\boldsymbol{B}^{t+1}$), while the latter merely ensure that $\mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C}^{t+1}) \le \mathcal{F}(\boldsymbol{B}^t, \boldsymbol{C}^t)$ (similarly for $\mathcal{F}(\boldsymbol{B}^{t+1}, \boldsymbol{C}^{t+1})$).

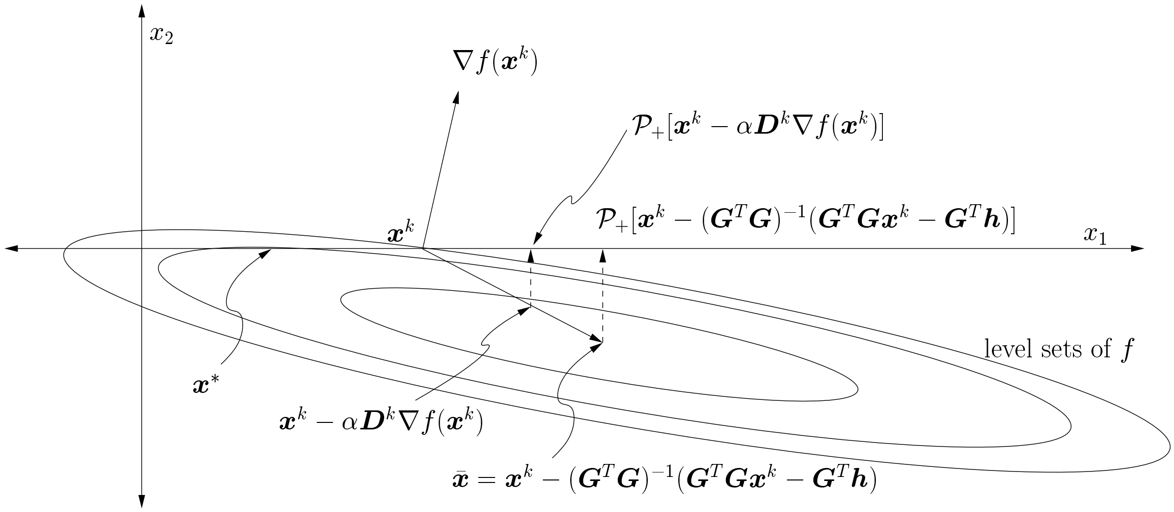
Since the Frobenius norm of a matrix is just the sum of Euclidean norms over columns (or rows), minimization or descent over either $\boldsymbol{B}$ or $\boldsymbol{C}$ boils down to solving a sequence of nonnegative least squares (NNLS) problems of the form

$$\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}} && f(\boldsymbol{x}) = \tfrac{1}{2}\|\boldsymbol{Gx}-\boldsymbol{h}\|_2^2, \\ &\text{subject to} && \boldsymbol{x} \ge \boldsymbol{0}. \end{aligned} \tag{2.1}$$

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 [13], FNNLS [5], and other procedures mentioned in [4]. However, as we show in [12], our approach to solving NNLS outperforms the other methods, hence we favor it as the method of choice for solving (2.1). 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 (1.2). 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 non-convexity of (1.2). 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 [18, 19], Paatero and Tapper [20] introduced a set of algorithms for NNMA and provided 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 (see [18]) for an actual implementation. Bierlaire et al. [3] developed a projected gradient method for NNLS, which Lin [16] applied to solve Problem (1.2). Recently, Merritt and Zhang [17] developed an interior-point gradient method—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 (2.1) 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 while retaining its algorithmic simplicity, and overcoming its deficiencies by employing a non-diagonal gradient scaling matrix.

The group of *inexact* methods has witnessed greater popularity and it includes Lee and Seung's [2000] multiplicative algorithms. Gonzalez and Zhang [10] proposed a variant of Lee and Seung's method that utilizes a different scaling scheme for negative gradients to get faster convergence speed. Berry et al. [1] report the Alternating Least Squares (ALS) procedure to be a simple but popular 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 §2.1. Another inexact approach is provided by the method



x_2
$\nabla f(\boldsymbol{x}^k)$
$\mathcal{P}_+[\boldsymbol{x}^k - \alpha \boldsymbol{D}^k \nabla f(\boldsymbol{x}^k)]$
$\mathcal{P}_+[\boldsymbol{x}^k - (\boldsymbol{G}^T\boldsymbol{G})^{-1}(\boldsymbol{G}^T\boldsymbol{G}\boldsymbol{x}^k - \boldsymbol{G}^T\boldsymbol{h})]$
$\boldsymbol{x}^k$
x_1
level sets of f
$\boldsymbol{x}^*$
$\boldsymbol{x}^k - \alpha \boldsymbol{D}^k \nabla f(\boldsymbol{x}^k)$
$\bar{\boldsymbol{x}} = \boldsymbol{x}^k - (\boldsymbol{G}^T\boldsymbol{G})^{-1}(\boldsymbol{G}^T\boldsymbol{G}\boldsymbol{x}^k - \boldsymbol{G}^T\boldsymbol{h})$

Figure 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 an approximation to the inverse of the Hessian. The current iterate is given as $\boldsymbol{x}^k$ and $I_+ = \{2\}$. Note that this setup reduces to the Newton-method for unconstrained cases (without projection), therefore we reach the 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.

of Zdunek and Cichocki [25] who proposed the combination of projection with a quasi-Newton procedure for NNMA, which we refer to as the ZC method.

Our FNMA$^{\mathrm{E}}$ procedure removes the theoretical deficiencies of both the ALS and ZC methods. As an additional alternative to these methods we present an inexact method called FNMA$^{\mathrm{I}}$ that shares the algorithmic framework with its exact counterpart FNMA$^{\mathrm{E}}$, and fixes the theoretical deficiencies of ALS and ZC while providing a computationally efficient procedure.

2.1 ALS and ZC Methods. As alluded to above, both the ALS and ZC methods have theoretical deficiencies, which can lead to non-monotonic 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 intimately 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 (2.1) 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 (2.1) may be written as

$$\boldsymbol{x} = \mathcal{P}_+[(\boldsymbol{G}^T\boldsymbol{G})^{-1}\boldsymbol{G}^T\boldsymbol{h}], \tag{2.2}$$

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

$$\boldsymbol{x}^{\text{new}} = \mathcal{P}_+[\boldsymbol{x}^{\text{old}} - \alpha\boldsymbol{D}(\boldsymbol{G}^T\boldsymbol{G}\boldsymbol{x}^{\text{old}} - \boldsymbol{G}^T\boldsymbol{h})], \tag{2.3}$$

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. Figure 1 illustrates why the updates (2.2) and (2.3) are inexact, wherein they fail to decrease the objective function for an arbitrary positive α. Observe that (2.2) essentially performs an exact-Newton step followed by projection, while (2.3) does 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 Figure 6). Our exact method, FNMA$^{\mathrm{E}}$, fixes this problem and is provably convergent unlike the ALS and ZC methods.

3 Algorithms and Theory

In this section we develop an algorithm and associated supporting theory for solving (1.2). An efficient solution of the NNLS subproblem (2.1) forms the core of FNMA$^{\mathrm{E}}$. 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 analogue of steepest descent, suffers from deficiencies such as slow convergence and zigzagging. For *unconstrained* optimization problems, it is known that the use of non-diagonal positive definite gradient scaling matrices alleviates such problems, and Bertsekas [2] developed a projection framework based on the Newton-method in that context. We build on that idea and employ non-diagonal gradient scaling based on the Quasi-Newton method for Problem (2.1), 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 (2.1) 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.,

$$I_+ = \{i \mid x_i^k = 0,\, [\nabla f(\boldsymbol{x}^k)]_i > 0\}. \tag{3.1}$$

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

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 $\boldsymbol{y}$ as follows

$$\boldsymbol{y} = \mathcal{P}_+\big[\boldsymbol{y}^k - \alpha \bar{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)\big], \tag{3.2}$$

where $\alpha \geq 0$, and $\bar{\boldsymbol{D}}^k$ 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 $\bar{\boldsymbol{D}}^k$ is a corresponding restricted scaling matrix.

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

$$\boldsymbol{x}^{k+1} \leftarrow \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{z}^k \end{bmatrix} = \begin{bmatrix} \mathcal{P}_+\big[\boldsymbol{y}^k - \alpha \bar{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)\big] \\ \boldsymbol{0} \end{bmatrix}, \tag{3.3}$$

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 $\boldsymbol{y}^{k+1}$.

In fact, any algorithm that finds $\boldsymbol{y}$ such that

$$g^k(\boldsymbol{y}) < g^k(\boldsymbol{y}^k), \quad \boldsymbol{y} \geq 0, \tag{3.4}$$

where

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

can be used to update $\boldsymbol{x}^k$ in (3.3), but since (3.4) is again a constrained problem, (3.2) remains a good choice for feasibility and efficiency of the overall algorithm. Furthermore, due to the resemblance of (3.2) 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 $\bar{\boldsymbol{D}}^k$ 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 $\bar{\boldsymbol{D}}^k$ 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, $\bar{\boldsymbol{D}}^k$ can be approximated by taking a proper sub-matrix of $\boldsymbol{D}^k$, which contains curvature information from the vectors $\{\boldsymbol{x}^k\}$. Based on 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 $\bar{\boldsymbol{D}}^k$ 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 choose the well-established BFGS method [6, 8, 9, 22] which incrementally approximates the inverse of the Hessian using only gradient information at each iteration.

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

$$\boldsymbol{H}^{k+1} = \boldsymbol{H}^k - \frac{\boldsymbol{H}^k \boldsymbol{u}\boldsymbol{u}^T \boldsymbol{H}^k}{\boldsymbol{u}^T \boldsymbol{H}^k \boldsymbol{u}} + \frac{\boldsymbol{w}\boldsymbol{w}^T}{\boldsymbol{u}^T \boldsymbol{w}}, \tag{3.5}$$

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 (3.5) yields

$$\begin{aligned} \boldsymbol{D}^{k+1} &= \boldsymbol{D}^k - \frac{(\boldsymbol{D}^k \boldsymbol{w}\boldsymbol{u}^T + \boldsymbol{u}\boldsymbol{w}^T \boldsymbol{D}^k)}{\boldsymbol{u}^T \boldsymbol{w}} \\ &\quad + \left(1 + \frac{\boldsymbol{w}^T \boldsymbol{D}^k \boldsymbol{w}}{\boldsymbol{u}^T \boldsymbol{w}}\right) \frac{\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T \boldsymbol{w}}. \end{aligned} \tag{3.6}$$

Since, $\nabla f(\boldsymbol{x}^k) = \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{x}^k - \boldsymbol{G}^T \boldsymbol{h}$ for the NNLS problem, (3.6) can be rewritten as

$$\begin{aligned} \boldsymbol{D}^{k+1} &= \boldsymbol{D}^k - \frac{(\boldsymbol{D}^k \boldsymbol{G}^T \boldsymbol{G}\boldsymbol{u}\boldsymbol{u}^T + \boldsymbol{u}\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{D}^k)}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{u}} \\ &\quad + \left(1 + \frac{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{D}^k \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{u}}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{u}}\right) \frac{\boldsymbol{u}\boldsymbol{u}^T}{\boldsymbol{u}^T \boldsymbol{G}^T \boldsymbol{G} \boldsymbol{u}}. \end{aligned} \tag{3.7}$$

Remark: Note that $\|\boldsymbol{G}\boldsymbol{u}\|_2^2$ appears as the denominator in the last two terms of (3.7). When $\boldsymbol{G}$ is of full-rank, (3.7) 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 (3.7) 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 definite, which can be easily satisfied by simply setting $\boldsymbol{D}^{k+1} = \boldsymbol{D}^k$. However, in practice (3.7) remains well-defined, and we do not usually encounter $\|\boldsymbol{G}\boldsymbol{u}\| = 0$.

Line-search. From (3.2) 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 [12].

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 (2.1). The main result of this section is the following theorem.

Theorem 3.1 (Convergence and Optimality)**.** *If $\boldsymbol{G}$ is of full-rank and $\{\boldsymbol{x}^k\}$ is the sequence of points generated by* (3.3)*, then every limit point of $\{\boldsymbol{x}^k\}$ is a stationary point of Problem* (2.1)*, and hence optimal since (2.1) 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 updates (3.2) ensure a monotonic descent in the objective function value (Lemma 3.1). Then, we show that the resulting sequence of iterates $\{\boldsymbol{x}^k\}$ has a limit-point (Lemma 3.2). Finally, we show that any limit point of the sequence $\{\boldsymbol{x}^k\}$ is also a stationary or KKT point of (2.1), thereby concluding the proof.

Lemma 3.1 (Descent)**.** *If $\boldsymbol{x}^k$ is not a stationary point of* (2.1)*, then there exists some constant $\bar{\alpha}$ such that*

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

where $\boldsymbol{x}^{k+1}$ is given by (3.3)*.*

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

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

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

$$y_i^k \neq 0 \quad \text{and} \quad [\nabla f(\boldsymbol{y}^k)]_i < 0.$$

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

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

where $\bar{\boldsymbol{D}}^{\boldsymbol{k}}$ 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 feasible descent direction. Thus, there exists $\alpha_1 > 0$ such that

$$g^k(\boldsymbol{y}^k - \alpha \bar{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k)) < g^k(\boldsymbol{y}^k), \quad \forall \alpha \in (0, \alpha_1],$$

Now let

$$\boldsymbol{\gamma}(\alpha) = \boldsymbol{y}^k - \alpha \bar{\boldsymbol{D}}^k \nabla f(\boldsymbol{y}^k).$$

Since g^k is continuous, there exists a $\delta > 0$ such that

$$\left\|\boldsymbol{\gamma}(\alpha) - \mathcal{P}_+[\boldsymbol{\gamma}(\alpha)]\right\|_2 < \delta,$$

which implies

$$\left\|g^k(\boldsymbol{\gamma}(\alpha)) - g^k\big(\mathcal{P}_+[\boldsymbol{\gamma}(\alpha)]\big)\right\|_2 < \tfrac{1}{2}\epsilon,$$

for any $\epsilon > 0$. Also note that there exists an $\alpha_2 > 0$ such that

$$\left\|\boldsymbol{\gamma}(\alpha) - \mathcal{P}_+[\boldsymbol{\gamma}(\alpha)]\right\|_2 < \delta, \quad \forall \alpha \in (0, \alpha_2],$$

for an arbitrary $\delta > 0$. Hence, by letting

$$\epsilon = \left\|g^k(\boldsymbol{\gamma}(\alpha_1)) - g^k(\boldsymbol{y}^k)\right\|_2,$$

we can conclude that there exists some $\bar{\alpha} = \min\{\alpha_1, \alpha_2\}$ such that

$$g^k\big(\mathcal{P}_+[\boldsymbol{\gamma}(\alpha)]\big) = g^k(\boldsymbol{y}) < g^k(\boldsymbol{y}^k), \quad \forall \alpha \in (0, \bar{\alpha}].$$

Since $\boldsymbol{z}^k$ remains fixed in $\boldsymbol{x}^{k+1}$, we conclude that

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

Lemma 3.2 (Limit point)**.** *Let $\{\boldsymbol{x}^k\}$ be a sequence of points generated by* (3.3)*. Then, this sequence has a limit point.*

Proof. Assume that we start the iteration with $\boldsymbol{x}^0$, where

$$\text{(3.8)} \qquad \boldsymbol{x}^0 \neq \boldsymbol{0}, \quad \text{and} \quad f(\boldsymbol{x}^0) = M,$$

and the M-level set of f includes $\boldsymbol{0}$ in its interior. i.e. $f(\boldsymbol{0}) = m < M$. By Lemma 3.1, $\{f(\boldsymbol{x}^k)\}$ 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. Consider the set $\mathbb{X}$ representing the intersection of the nonnegative orthant with the M-level set of f and 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 by $\boldsymbol{0}$ and $\boldsymbol{u}$ hence it has a limit point in $\mathbb{X}$. This concludes the proof of the lemma. $\square$

Assumption 3.1. Let $\{\boldsymbol{x}^k\}$ be a sequence generated by (3.2) and (3.3). Then, for any subsequence $\{\boldsymbol{x}^k\}_{k\in\mathcal{K}}$ that converges to a nonstationary point,

$$\limsup_{k\to\infty,k\in\mathcal{K}} \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\| < \infty,$$

$$\limsup_{k\to\infty,k\in\mathcal{K}} \nabla f(\boldsymbol{x}^k)^T(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k) < 0. \tag{3.9}$$

Assumption 3.1 is known as *gradient related condition* in optimization literature and plays crucial role to prove the convergence of a number of methods. Though we take this assumption as a given in this paper, it can be shown that our method actually satisfies these conditions [12].

Finally, we present a proof for our main theorem 3.1.

Proof. Assume $\{\boldsymbol{x}^k\}$ converges to a nonstationary point $\bar{\boldsymbol{x}}$. From lemma 3.1, it can be shown that there exists some ϵ_k such that

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

and

$$\lim_{k\to\infty} \epsilon_k = 0.$$

Since f is continuous over $\mathbb{X}$ and by lemma 3.1 and 3.2,

$$\lim_{k\to\infty} f(\boldsymbol{x}^k) = f(\bar{\boldsymbol{x}}).$$

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 (3.9) in assumption 3.1. □

3.3 FNMA$^{\text{E}}$: an *exact* method for NNMA. Now we extend the ideas from §3.1 to the matrix case. To that end, we need to redefine quantities in terms of matrices. First, we compute the gradient matrices $\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C})$ and $\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C})$ as follows

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

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

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

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

$$\big[\mathcal{Z}_+[\boldsymbol{X}]\big]_{ij} = \begin{cases} X_{ij}, & (i,j) \notin I_+, \\ 0, & \text{otherwise.} \end{cases} \tag{3.10}$$

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

```
Input: $\boldsymbol{A} \in \mathbb{R}_+^{M\times N}$,  $K \in N$
Output: $\boldsymbol{B} \in \mathbb{R}_+^{M\times K}$, $\boldsymbol{C} \in \mathbb{R}_+^{K\times N}$
1. Initialize $\boldsymbol{B}^0, \boldsymbol{C}^0, t = 0$.
repeat
  2. $\boldsymbol{B} \leftarrow \boldsymbol{B}^t$,  $\boldsymbol{C}^{\text{old}} \leftarrow \boldsymbol{C}^t$.
  repeat
    3.1. Compute the gradient matrix $\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}^{old})$.
    3.2. Compute fixed set $I_+$ for $\boldsymbol{C}^{old}$.
    3.3. Compute the step length vector $\boldsymbol{\alpha}$.
    3.4. Update $\boldsymbol{C}^{old}$ as
      $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B};\boldsymbol{C}^{\text{old}})\big]$;  $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\boldsymbol{D}\boldsymbol{U}\big]$
      $\boldsymbol{C}^{\text{new}} \leftarrow \mathcal{P}_+\big[\boldsymbol{C}^{\text{old}} - \boldsymbol{U}\cdot\text{diag}(\boldsymbol{\alpha})\big]$
    3.5. $\boldsymbol{C}^{\text{old}} \leftarrow \boldsymbol{C}^{\text{new}}$.
    3.6. Update $\boldsymbol{D}$ if necessary.
  until $\boldsymbol{C}^{\text{old}}$ converges
  4. $\boldsymbol{C}^{t+1} \leftarrow \boldsymbol{C}^{\text{old}}$.
  5. $\boldsymbol{C} \leftarrow \boldsymbol{C}^{t+1}$,  $\boldsymbol{B}^{\text{old}} \leftarrow \boldsymbol{B}^t$.
  repeat
    6.1. Compute the gradient matrix $\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B}^{\text{old}};\boldsymbol{C})$.
    6.2. Compute fixed set $I_+$ for $\boldsymbol{B}^{\text{old}}$.
    6.3. Compute the step length vector $\boldsymbol{\alpha}$.
    6.4. Update $\boldsymbol{B}^{old}$ as:
      $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B}^{\text{old}};\boldsymbol{C})\big]$;  $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\boldsymbol{U}\boldsymbol{D}\big]$
      $\boldsymbol{B}^{\text{new}} \leftarrow \mathcal{P}_+\big[\boldsymbol{B}^{\text{old}} - \text{diag}(\boldsymbol{\alpha})\cdot\boldsymbol{U}\big]$
    6.5. $\boldsymbol{B}^{\text{old}} \leftarrow \boldsymbol{B}^{\text{new}}$.
    6.6. Update $\boldsymbol{D}$ if necessary.
  until $\boldsymbol{B}^{\text{old}}$ converges
  7. $\boldsymbol{B}^{t+1} \leftarrow \boldsymbol{B}^{\text{old}}$.
  8. $t \leftarrow t + 1$.
until Stopping criteria are met
```

Now we have all the pieces to describe the overall algorithm for solving the NNMA problem (1.2). Algorithm 1 presents our proposed method which we name Fast Nonnegative Matrix Approximation - exact (FNMA$^{\text{E}}$).

In Steps 3.4 and 6.4 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 formulation as follows. In Problem (2.1), a series of BFGS updates for $\boldsymbol{D}$ aim at estimating the inverse of Hessian. In Problem (1.2), a matrix-wise extension of NNLS, every column shares the same Hessian, whereby each column can also share the approximation of the inverse of the 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.

Theorem 3.2 (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 the algorithm FNMA^{E} converges to a stationary point of Problem* (1.2)*.*

Proof. Algorithm 1 essentially performs the following alternating minimization at each outer iteration

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

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

Similar to the argument in Lemma 3.2, the domain of Problem (1.2) 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, so that

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

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

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

Now we can invoke the proof of the two-block Gauss-Seidel method [11] to conclude our claim. □

3.4 FNMA$^{\mathbf{I}}$: 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 provide 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 non-diagonal 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 3.3 (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.*

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 $\tau$ 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:
         $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\nabla_{\boldsymbol{C}}\mathcal{F}(\boldsymbol{B}; \boldsymbol{C}^{\text{old}})\big]$;   $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\big(\boldsymbol{B}^T\boldsymbol{B}\big)^{-1}\boldsymbol{U}\big]$
         $\boldsymbol{C}^{\text{new}} \leftarrow \mathcal{P}_+\big[\boldsymbol{C}^{\text{old}} - \alpha\boldsymbol{U}\big]$
    3.4. $\boldsymbol{C}^{\text{old}} \leftarrow \boldsymbol{C}^{\text{new}}$.
  end for
  4. $\boldsymbol{C}^{t+1} \leftarrow \boldsymbol{C}^{\text{old}}$.
  5. $\boldsymbol{C} \leftarrow \boldsymbol{C}^{t+1}$,   $\boldsymbol{B}^{\text{old}} \leftarrow \boldsymbol{B}^k$.
  for $i = 1$ to $\tau$ do
    6.1. Compute the gradient matrix $\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B}^{\text{old}}; \boldsymbol{C})$.
    6.2. Compute fixed set $I_+$ for $\boldsymbol{B}^{\text{old}}$.
    6.3. Update $\boldsymbol{B}^{\text{old}}$ as:
         $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\nabla_{\boldsymbol{B}}\mathcal{F}(\boldsymbol{B}^{old}; \boldsymbol{C})\big]$;   $\boldsymbol{U} \leftarrow \mathcal{Z}_+\big[\boldsymbol{U}\big(\boldsymbol{C}\boldsymbol{C}^T\big)^{-1}\big]$
         $\boldsymbol{B}^{\text{new}} \leftarrow \mathcal{P}_+\big[\boldsymbol{B}^{\text{old}} - \alpha\boldsymbol{U}\big]$
    6.4. $\boldsymbol{B}^{old} \leftarrow \boldsymbol{B}^{new}$.
  end for
  7. $\boldsymbol{B}^{t+1} \leftarrow \boldsymbol{B}^{old}$.
  8. $t \leftarrow t + 1$.
until Stopping criteria are met
```

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

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

where $\boldsymbol{\alpha}$ is computed by Step 3.3 or 6.3 from FNMA$^{\mathrm{E}}$, Lemma 3.1 ensures that Steps 2-3 decrease the objective monotonically. □

Remark 1: If any α_i becomes zero, then the inner loop for the current alternating step should be terminated to guarantee monotonicity.

Remark 2: 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. Assume Steps 3.3 and 6.3 from FNMA$^{\mathrm{I}}$ are in the form

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

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

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

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

3.5 Extensions to handle regularization. The regularized NNMA problem (1.3) 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 (2.1), which is also known as BLS [4].

$$\text{(3.11)} \qquad \begin{array}{ll} \underset{\boldsymbol{x}}{\text{minimize}} & \frac{1}{2}\|\boldsymbol{G}\boldsymbol{x} - \boldsymbol{h}\|^2, \\ \text{subject to} & \boldsymbol{p} \le \boldsymbol{x} \le \boldsymbol{q}. \end{array}$$

Problem (3.11) can be solved just as we solved (2.1). We need to modify the definition of the *fixed-set* (3.1) so that

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

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

$$\text{(3.12)} \qquad \mathcal{P}_\Omega[\boldsymbol{x}_i] = \begin{cases} \boldsymbol{p}_i & : \quad \boldsymbol{x}_i \le \boldsymbol{p}_i \\ \boldsymbol{x}_i & : \quad \boldsymbol{p}_i < \boldsymbol{x}_i < \boldsymbol{q}_i \\ \boldsymbol{q}_i & : \quad \boldsymbol{q}_i \le \boldsymbol{x}_i \end{cases}$$

Given these definitions, it is easy to check that Lemma 3.1 holds without significant modification and Theorem 3.1 also follows under the same Assumption 3.1. The fact that the domain is a compact set for (3.11), obviates the need for Lemma 3.2 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 (1.4). We omit the details for brevity, noting that the modifications needed are minor, for example, the fixed set for $\boldsymbol{C}$ is redefined as

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

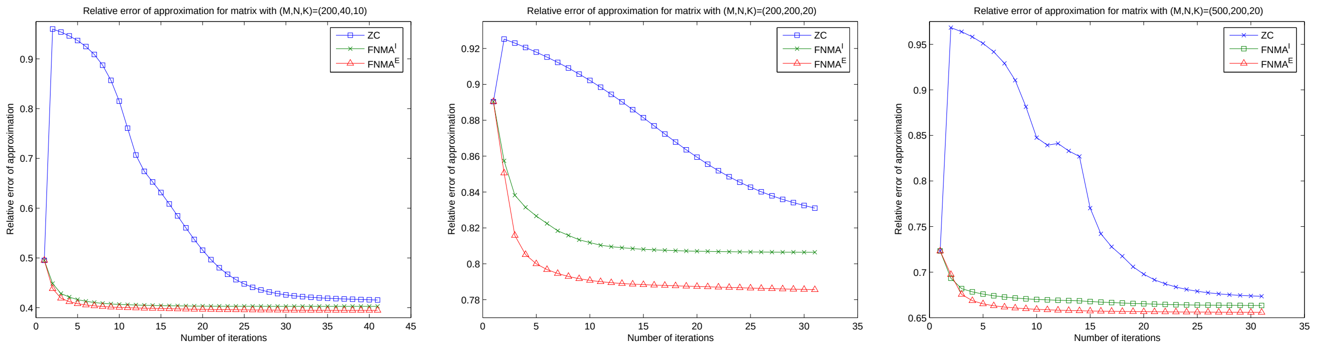
By taking a projection step similar to (3.12) we can construct the desired method (also see Projection (3.10)).

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 Lee & Seung (LS) method [14], Zdunek and Cichocki's (ZC) (2006) method, and the ALS approach for solving the least-squares NNMA problem. We show results on random dense matrices (§4.1), real-world sparse matrices (§4.2), and real-world dense data (§4.3). Our experiments show that FNMA$^{\mathrm{E}}$ and FNMA$^{\mathrm{I}}$ produce better numerical results than 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 [7]. 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 [23], 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 text analysis in §4.2, and on image processing in §4.3.

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., $\|\boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}\|_{\mathrm{F}}/\|\boldsymbol{A}\|_{\mathrm{F}}$ 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



(a) Dense

(b) Sparse

(c) Sparse

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

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^E and FNMA^I against ZC on three data matrices. The results are reported in Figure 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 Figure 2(b).

4.1.2 Comparisons against LS and ALS. On a larger matrix, Figure 3 shows a comparison of the approximation accuracies for LS, ALS, FNMA^E, and FNMA^I. We see that FNMA^E achieves the best objective function values of all the methods presented. However, FNMA^E can take more running time than the other methods because of its *exact* nature. Therefore, FNMA^E is to be preferred when reconstruction accuracy is more important, while FNMA^I is recommended when running time is more important. We now present two more experiments to highlight the

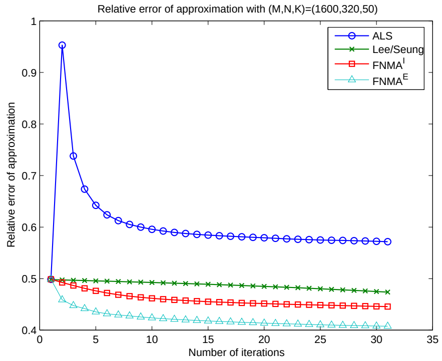


Figure 3: Relative error values against iteration count for a random dense matrix of size 1600 × 320 for a rank 50 approximation. All methods other than ALS show a monotonic decrease when initialized with one step of LS.

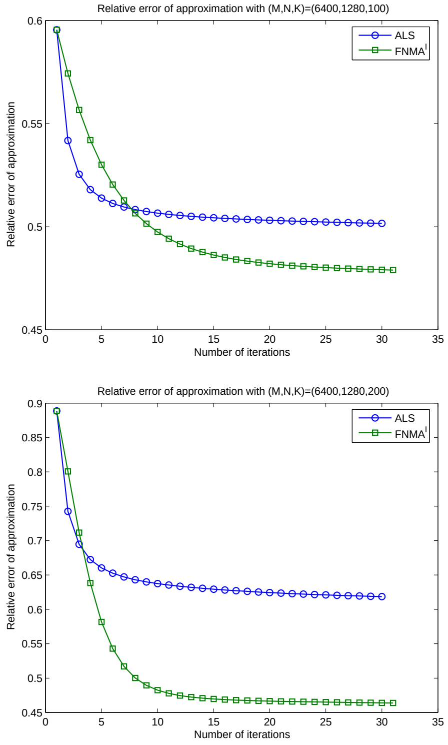
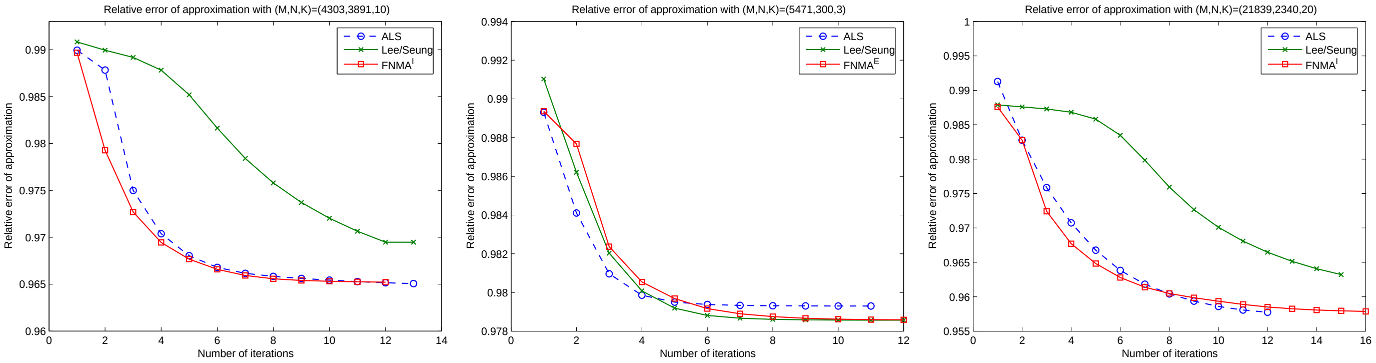


Figure 4: Relative error values against iteration count for a random dense matrix of size 6400×1280 for a rank 100 (top) and rank 200 approximation (bottom).



Classic3 (rank 10) Classic300 (rank 3) Yahoo-News (rank 20)

Figure 5: Relative errors obtained by ALS, LS, and FNMA$^{\mathrm{I}}$ on the Classic3, Classic300, and Yahoo-News datasets. Observe the range of the errors on the y-axes of all the plots. Even though there seems to be a difference amongst the three algorithms, they all perform essentially the same, since the rank of the approximation sought in each problem is very small relative to the dimensionality of the data. ALS and FNMA$^{\mathrm{I}}$ behave very similarly, both of them slightly better than LS.

advantages of FNMA$^{\mathrm{I}}$ over ALS, which owing to its *ad-hoc* nature leads to inferior accuracies (see also §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 (Figure 4), the accuracy achieved by FNMA$^{\mathrm{I}}$ is 25% higher than that achieved by ALS.

4.2 Application to Text Analysis. Owing to its ability to produce sparse representations, NNMA has been applied to text analysis, for example, see [15, 21, 24]. We show results of running ALS, LS and FNMA$^{\mathrm{I}}$ on three text datasets, which are high-dimensional and sparse. These datasets are

- Classic3: A corpus containing 3891 documents drawn from the areas of information retrieval (CISI), aeronautical systems (CRAN), and medical journal articles (MED). After standard pre-processing, the dataset resulted in a 4303×3891 matrix.

- Classic300: A subset of 300 documents taken from Classic3, with 100 randomly chosen documents from each of the three categories given above. This data matrix has size 5471×300.

- Yahoo News (K-Series): This corpus consists of 2340 news articles belonging to 20 different categories. The size of this data matrix is 21819×2340.

Figure 5 illustrates the objective function values achieved by running ALS, LS and FNMA$^{\mathrm{I}}$ on these text datasets. The rank of the decomposition was picked to be small, since that was enough to separate the clusters inherent in the data. All the algorithms seem to perform equally well, with marginal differences in their final objective function values. We infer that this is an outcome of the small rank of the approximation. As the rank increases, all the three algorithms encounter numerical difficulties due to singularities or divisions by zero. To counter exactly this situation, the regularized version of NNMA can be used.

Table 1 shows the top keywords obtained from a rank-3 approximation to the Classic3 matrix, wherein the ten largest entries from each column of $\boldsymbol{B}$ are extracted, since columns of $\boldsymbol{B}$ can be interpreted as the basis vectors for documents (columns of matrix $\boldsymbol{A}$). From the keywords, it is quite easy to recognize that the three underlying categories are well represented.

Table 1: Top 10 keywords (per basis vector of $\boldsymbol{B}$) obtained by FNMA$^{\mathrm{I}}$ for a rank-3 approximation to the Classic3 dataset

CISI	CRAN	MED
retrieval	wing	patients
system	pressure	cells
systems	mach	growth
indexing	supersonic	hormone
scientific	shock	cancer
science	jet	treatment
index	lift	buckling
search	wings	blood
computer	body	cases
document	theory	cell

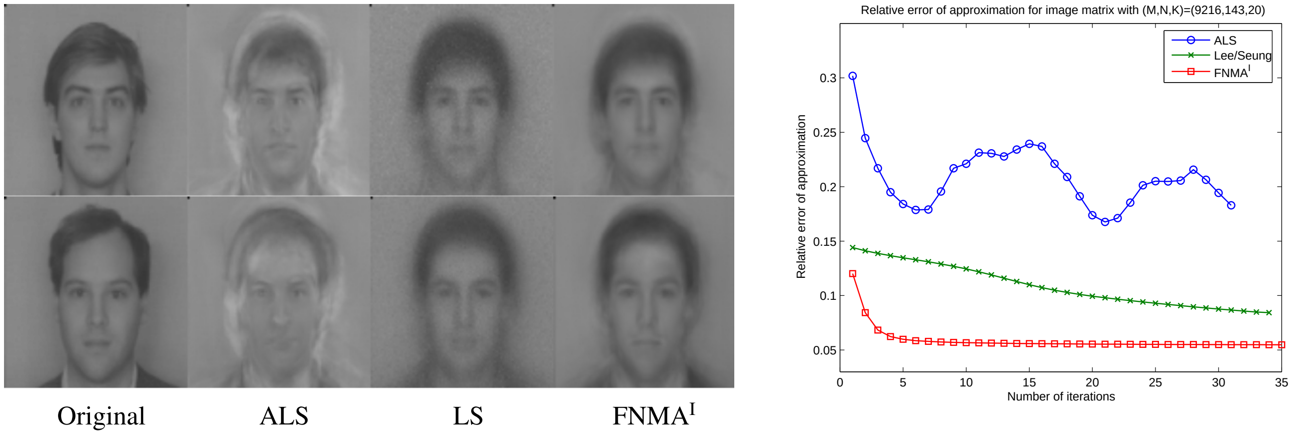


Figure 6: Image reconstruction as obtained by the ALS, LS, and FNMA$^{\mathrm{I}}$ procedures. The figure illustrates two randomly chosen images out of the 143 reconstructed images, each with 96 × 96 pixels. The reconstruction was computed from a rank-20 approximation to the input image matrix, which was of size 9216×143. The first image in each row is the original, followed by reconstructions obtained via ALS, LS, and FNMA$^{\mathrm{I}}$. From the images above, FNMA$^{\mathrm{I}}$ is seen to obtain the best reconstruction and the relative errors as plotted on the right attest to this observation. Note how the ALS leads to a non-monotonic change in the objective function value (as explained in §2.1).

We remark that due to random initializations, it can sometimes be the case that a rank-3 approximation does not cover the three different categories, and one can end up finding sub-categories of one of the bigger categories. This is a well-known problem with many topic-discovery systems. However, in an exploratory mining system, one can always request a higher rank approximation, and usually for the Classic3 dataset, a rank-4 approximation reveals all three underlying categories. NNMA can be used as a topic discovery and analysis system, especially due to the fact that it yields a nonnegative decomposition of the input data, and text-data is inherently nonnegative, whereby the resulting decomposition is easy to interpret (as shown in Table 1).

4.3 Application to Image processing. NNMA was originally motivated by Lee and Seung [15] using an image processing application. Many other authors have also considered NNMA for image processing, graphics, or face recognition applications. 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 Figure 6, which shows accuracies for a rank-20 approximation to a 9216×143 matrix of face images.[1]

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$^{\mathrm{I}}$ achieves the best objective values of all three algorithms compared, and a corresponding better reconstruction is observed (Figure 6).

5 Conclusions

In this paper, we have presented new and improved Newton-type methods for the least-squares NNMA problem. By employing a non-diagonal gradient scaling scheme, our algorithms use curvature information and thus overcome deficiencies of gradient descent based methods. Our methods also rectify serious drawbacks in existing methods such as alternating least squares and Zdunek and Cichocki's 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$^{\mathrm{E}}$, which shows good performance in terms of approximation accuracy, is suitable for applications that require superior accuracy. Our inexact implementation FNMA$^{\mathrm{I}}$ is more suitable for applications that are more constrained by computational efficiency rather than accuracy.

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. *Computa-*

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

tional Statistics and Data Analysis, 2006. Preprint.

[2] D. P. Bertsekas. Projected Newton Methods for Optimization Problems with Simple Constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

[3] M. Bierlaire, Philipe L. Toint, and D. Tuyttens. On Iterative Algorithms for Linear Least Squares Problems with Bound constraints. *Linear Algebra and its Applications*, 143:111–143, 1991.

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

[5] R. Bro and S. D. Jong. A Fast Non-negativity-constrained Least Squares Algorithm. *Journal of Chemometrics*, 11(5):393–401, 1997.

[6] C. G. Broyden. The Convergence of A Class of Double-rank Minimization Algorithms, Part I. *Journal of the Institute of Mathematics and Its Applications*, 6:76–90, 1970.

[7] A. Cichocki and R. Zdunek. NMFLAB – MATLAB Toolbox for Non-Negative Matrix Factorization. Online, 2006.

[8] R. Fletcher. A New Approach to Variable Metric Algorithms. *Computer Journal*, pages 317–322, 1970.

[9] D. Goldfarb. A Family of Variable-metric Methods Derived by Variational Means. *Mathematics of Computation*, 24(109):23–26, 1970.

[10] E. F. Gonzalez and Y. Zhang. Accelerating The Lee-Seung Algorithm for Nonnegative Matrix Factorization. Technical Report TR-05-02, Rice University, 2005.

[11] L. Grippo and M. Sciandrone. On The Convergence of The Block Nonlinear Gauss-Seidel Method under Convex Constraints. *Operations Research Letters*, 26: 127–136, 2000.

[12] 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 Univ. of Texas at Austin, 2006.

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

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

[15] D. D. Lee and H. S. Seung. Learning The Parts of Objects by Nonnegative Matrix Factorization. *Nature*, 401:788–791, 1999.

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

[17] M. Merritt and Y. Zhang. Interior-Point Gradient Method for Large-Scale Totally Nonnegative Least Squares Problems. *Journal of Optimization Theory and Applications*, 126(1):191–202, 2005.

[18] P. Paatero. Least-squares Formulation of Robust Non-negative Factor Analysis. *Chemometrics and Intelligent Laboratory Systems*, 37:23–35, 1997.

[19] P. Paatero. The Multilinear Engine—A Table-driven Least Squares Program for Solving Multilinear Problems, Including The N-way Parallel Factor Analysis Model. *Journal of Computational and Graphical Statistics*, 8(4):854–888, 1999.

[20] P. Paatero and U. Tapper. Positive Matrix Factorization: A Nonnegative Factor Model with Optimal Utilization of Error Estimates of Data Values. *Environmetrics*, 5 (111–126), 1994.

[21] F. Shahnaz, M. Berry, P. Pauca, and R. Plemmons. Document Clustering using Nonnegative Matrix Factorization. *Journal on Information Processing and Management*, 42:373–386, 2006.

[22] D. F. Shanno. Conditioning of Quasi-Newton Methods for Function Minimization. *Mathematics of Computation*, 24(111):647–656, 1970.

[23] 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.

[24] W. Xu, X. Liu, and Y. Gong. Document Clustering Based on Nonnegative Matrix Factorization. In *SIGIR'03*, pages 267–273, 2003.

[25] R. Zdunek and A. Cichocki. Non-Negative Matrix Factorization with Quasi-Newton Optimization. In *Eighth International Conference on Artificial Intelligence and Soft Computing, ICAISC*, pages 870–879, 2006.