Journal of Machine Learning Research 15 (2014) 1011-1039 Submitted 9/13; Published 3/14

Ellipsoidal Rounding for Nonnegative Matrix Factorization Under Noisy Separability

Tomohiko Mizutani MIZUTANI@KANAGAWA-U.AC.JP
Department of Information Systems Creation
Kanagawa University
Yokohama, 221-8686, Japan

Editor: Nathan Srebro

Abstract

We present a numerical algorithm for nonnegative matrix factorization (NMF) problems under noisy separability. An NMF problem under separability can be stated as one of finding all vertices of the convex hull of data points. The research interest of this paper is to find the vectors as close to the vertices as possible in a situation in which noise is added to the data points. Our algorithm is designed to capture the shape of the convex hull of data points by using its enclosing ellipsoid. We show that the algorithm has correctness and robustness properties from theoretical and practical perspectives; correctness here means that if the data points do not contain any noise, the algorithm can find the vertices of their convex hull; robustness means that if the data points contain noise, the algorithm can find the near-vertices. Finally, we apply the algorithm to document clustering, and report the experimental results.

Keywords: nonnegative matrix factorization, separability, robustness to noise, enclosing ellipsoid, document clustering

1. Introduction

This paper presents a numerical algorithm for nonnegative matrix factorization (NMF) problems under noisy separability. The problem can be regarded as a special case of an NMF problem. Let $\mathbb{R}_+^{d\times m}$ be the set of d-by-m nonnegative matrices, and $\mathbb{N}$ be the set of nonnegative integer numbers. A nonnegative matrix is a real matrix whose elements are all nonnegative. For a given $\boldsymbol{A} \in \mathbb{R}_+^{d\times m}$ and $r \in \mathbb{N}$, the nonnegative matrix factorization (NMF) problem is to find $\boldsymbol{F} \in \mathbb{R}_+^{d\times r}$ and $\boldsymbol{W} \in \mathbb{R}_+^{r\times m}$ such that the product $\boldsymbol{FW}$ is as close to $\boldsymbol{A}$ as possible. The nonnegative matrices $\boldsymbol{F}$ and $\boldsymbol{W}$ give a factorization of $\boldsymbol{A}$ of the form,

$$\boldsymbol{A} = \boldsymbol{FW} + \boldsymbol{N},$$

where $\boldsymbol{N}$ is a d-by-m matrix. This factorization is referred to as the NMF of $\boldsymbol{A}$.

Recent studies have shown that NMFs are useful for tackling various problems such as facial image analysis (Lee and Seung, 1999), topic modeling (Arora et al., 2012b, 2013; Ding et al., 2013), document clustering (Xu et al., 2003; Shahnaz et al., 2006), hyperspectral unmixing (Nascimento and Dias, 2005; Miao and Qi, 2007; Gillis and Vavasis, 2014), and blind source separation (Cichocki et al., 2009). Many algorithms have been developed in the context of solving such practical applications. However, there are some drawbacks in

the use of NMFs for such applications. One of them is in the hardness of solving an NMF problem. In fact, the problem has been shown to be NP-hard by Vavasis (2009).

As a remedy for the hardness of the problem, Arora et al. (2012a) proposed to exploit the notion of separability, which was originally introduced by Donoho and Stodden (2003) for the uniqueness of NMF. An NMF problem under separability becomes a tractable one. *Separability* assumes that $\boldsymbol{A} \in \mathbb{R}_+^{d\times m}$ can be represented as

$$\boldsymbol{A} = \boldsymbol{F}\boldsymbol{W} \text{ for } \boldsymbol{F} \in \mathbb{R}_+^{d\times r} \text{ and } \boldsymbol{W} = (\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi} \in \mathbb{R}_+^{r\times m}, \tag{1}$$

where $\boldsymbol{I}$ is an r-by-r identity matrix, $\boldsymbol{K}$ is an r-by-$(m-r)$ nonnegative matrix, and $\boldsymbol{\Pi}$ is an m-by-m permutation matrix. This means that each column of $\boldsymbol{F}$ corresponds to that of $\boldsymbol{A}$ up to a scaling factor. A matrix $\boldsymbol{A}$ is said to be a *separable matrix* if it can be represented in the form (1). In this paper, we call $\boldsymbol{F}$ the *basis matrix* of a separable matrix, and $\boldsymbol{W}$, as well as its submatrix $\boldsymbol{K}$, the *weight matrix*. *Noisy separability* assumes that a separable matrix $\boldsymbol{A}$ contains a noise matrix $\boldsymbol{N}$ such that $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{N}$, where $\boldsymbol{N}$ is a d-by-m matrix. Arora et al. showed that there exists an algorithm for finding the near-basis matrix of a noisy separable one if the noise is small in magnitude. Although a separability assumption restricts the fields of application for NMFs, it is known to be reasonable at least, in the contexts of document clustering (Kumar et al., 2013), topic modeling (Arora et al., 2012a,b, 2013), and hyperspectral unmixing (Gillis and Vavasis, 2014). In particular, this assumption is widely used as a pure-pixel assumption in hyperspectral unmixing (see, for instance, Nascimento and Dias, 2005; Miao and Qi, 2007; Gillis and Vavasis, 2014).

An NMF problem under noisy separability is to seek for the basis matrix of a noisy separable one. The problem is formally described as follows:

Problem 1 *Let a data matrix $\boldsymbol{M}$ be a noisy separable matrix of size d-by-m. Find an index set $\mathcal{I}$ with cardinality r on $\{1,\dots,m\}$ such that $\boldsymbol{M}(\mathcal{I})$ is as close to the basis matrix $\boldsymbol{F}$ as possible.*

Here, $\boldsymbol{M}(\mathcal{I})$ denotes a submatrix of $\boldsymbol{M}$ that consists of every column vector with an index in $\mathcal{I}$. We call the column vector of $\boldsymbol{M}$ a *data point* and that of the basis matrix $\boldsymbol{F}$ a *basis vector*. An ideal algorithm for the problem should have correctness and robustness properties; correctness here means that, if the data matrix $\boldsymbol{M}$ is just a separable one, the algorithm can find the basis matrix; robustness means that, if the data matrix $\boldsymbol{M}$ is a noisy separable one, the algorithm can find the near-basis matrix. A formal description of the properties is given in Section 2.1

We present a novel algorithm for Problem 1. The main contribution of this paper is to show that the algorithm has correctness and robustness properties from theoretical and practical perspectives. It is designed on the basis of the geometry of a separable matrix. Under reasonable assumptions, the convex hull of the column vectors of a separable matrix forms a simplex, and in particular, the basis vectors correspond to the vertices. Therefore, if all vertices of a simplex can be found, we can obtain the basis matrix of the separable matrix. Our algorithm uses the fact that the vertices of simplex can be found by an ellipsoid. That is, if we draw the minimum-volume enclosing ellipsoid (MVEE) for a simplex, the ellipsoid only touches its vertices. More precisely, we give plus and minus signs to the vertices of a simplex, and take the convex hull; it becomes a crosspolytope having the simplex as one of

the facets. Then, the MVEE for the crosspolytope only touches the vertices of the simplex with plus and minus signs.

Consider Problem 1 without noise. In this case, the data matrix is just a separable one. Our algorithm computes the MVEE for the data points and outputs the points on the boundary of the ellipsoid. Then, the obtained points correspond to the basis vectors for a separable matrix. We show in Theorem 5 that the correctness property holds. Moreover, the algorithm works well even when the problem contains noise. We show in Theorem 9 that, if the noise is lower than a certain level, the algorithm correctly identifies the near-basis vectors for a noisy separable matrix, and hence, the robustness property holds. The existing algorithms (Arora et al., 2012a; Bittorf et al., 2012; Gillis, 2013; Gillis and Luce, 2013; Gillis and Vavasis, 2014; Kumar et al., 2013) are formally shown to have these correctness and robustness properties. In Section 2.4, our correctness and robustness properties are compared with those of the existing algorithms.

It is possible that noise will exceed the level that Theorem 9 guarantees. In such a situation, the MVEE for the data points may touch many points. Hence, r points need to be selected from the points on the boundary of the ellipsoid. We make the selection by using existing algorithms such as SPA (Gillis and Vavasis, 2014) and XRAY (Kumar et al., 2013). Our algorithm thus works as a preprocessor which filters out basis vector candidates from the data points and enhance the performance of existing algorithms.

We demonstrated the robustness of the algorithms to noise through experiments with synthetic data sets. In particular, we experimentally compared our algorithm with SPA and XRAY. We synthetically generated data sets with various noise levels, and measured the robustness of an algorithm by its recovery rate. The experimental results indicated that our algorithm can improve the recovery rates of SPA and XRAY.

Finally, we applied our algorithm to document clustering. Separability for a document-term matrix means that each topic has an anchor word. An anchor word is a word which is contained in one topic but not contained in the other topics. If an anchor word is found, it suggests the existence of its associated topic. We conducted experiments with document corpora and compared the clustering performances of our algorithm and SPA. The experimental results indicated that our algorithm would usually outperform SPA and can extract more recognizable topics.

The rest of this paper is organized as follows. Section 2 gives an outline of our algorithm and reviews related work. Then, the correctness and robustness properties of our algorithm are given, and a comparison with existing algorithms is described. Section 3 reviews the formulation and algorithm of computing the MVEE for a set of points. Sections 4 and 5 are the main part of this paper. We show the correctness and robustness properties of our algorithm in Section 4, and discuss its practical implementation in Section 5. Section 6 reports the numerical experiments for the robustness of algorithms and document clustering. Section 7 gives concluding remarks.

1.1 Notation and Symbols

We use $\mathbb{R}^{d\times m}$ to denote a set of real matrices of size d-by-m, and $\mathbb{R}_+^{d\times m}$ to denote a set of nonnegative matrices of d-by-m. Let $\boldsymbol{A} \in \mathbb{R}^{d\times m}$. The symbols $\boldsymbol{A}^\top$ and $\mathrm{rank}(\boldsymbol{A})$ respectively denote the transposition and the rank. The symbols $||\boldsymbol{A}||_p$ and $||\boldsymbol{A}||_F$ are the matrix p-norm

and the Frobenius norm. The symbol $\sigma_i(\boldsymbol{A})$ is the ith largest singular value. Let $\boldsymbol{a}_i$ be the ith column vector of $\boldsymbol{A}$, and $\mathcal{I}$ be a subset of $\{1,\ldots,m\}$. The symbol $\boldsymbol{A}(\mathcal{I})$ denotes a d-by-$|\mathcal{I}|$ submatrix of $\boldsymbol{A}$ such that $(\boldsymbol{a}_i : i \in \mathcal{I})$. The convex hull of all the column vectors of $\boldsymbol{A}$ is denoted by $\mathrm{conv}(\boldsymbol{A})$, and referred to as the convex hull of $\boldsymbol{A}$ for short. We denote an identity matrix and a vector of all ones by $\boldsymbol{I}$ and $\boldsymbol{e}$, respectively.

We use $\mathbb{S}^d$ to denote a set of real symmetric matrices of size d. Let $\boldsymbol{A} \in \mathbb{S}^d$. If the matrix is positive definite, we represent it as $\boldsymbol{A} \succ \boldsymbol{0}$. Let $\boldsymbol{A}_1 \in \mathbb{S}^d$ and $\boldsymbol{A}_2 \in \mathbb{S}^d$. We denote by $\langle \boldsymbol{A}_1, \boldsymbol{A}_2 \rangle$ the Frobenius inner product of the two matrices which is given as the trace of matrix $\boldsymbol{A}_1\boldsymbol{A}_2$.

We use a MATLAB-like notation. Let $\boldsymbol{A}_1 \in \mathbb{R}^{d\times m_1}$ and $\boldsymbol{A}_2 \in \mathbb{R}^{d\times m_2}$. We denote by $(\boldsymbol{A}_1, \boldsymbol{A}_2)$ the horizontal concatenation of the two matrices, which is a d-by-(m_1+m_2) matrix. Let $\boldsymbol{A}_1 \in \mathbb{R}^{d_1\times m}$ and $\boldsymbol{A}_2 \in \mathbb{R}^{d_2\times m}$. We denote by $(\boldsymbol{A}_1; \boldsymbol{A}_2)$ the vertical concatenation of the two matrices, and it is a matrix of the form,

$$\begin{pmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \end{pmatrix} \in \mathbb{R}^{(d_1+d_2)\times m}.$$

Let $\boldsymbol{A}$ be a d-by-m rectangular diagonal matrix having diagonal elements $a_1,\ldots,a_t$ where $t = \min\{d, m\}$. We use $\mathrm{diag}(a_1,\ldots,a_t)$ to denote the matrix.

2. Outline of Proposed Algorithm and Comparison with Existing Algorithms

Here, we formally describe the properties mentioned in Section 1 that an algorithm is expected to have, and also describe the assumptions we place on Problem 1. Next, we give a geometric interpretation of a separable matrix under these assumptions, and then, outline the proposed algorithm. After reviewing the related work, we describe the correctness and robustness properties of our algorithm and compare with those of the existing algorithms.

2.1 Preliminaries

Consider Problem 1 whose data matrix $\boldsymbol{M}$ is a noisy separable one of the form $\boldsymbol{A} + \boldsymbol{N}$. Here, $\boldsymbol{A}$ is a separable matrix of (1) and $\boldsymbol{N}$ is a noise matrix. We can rewrite it as

$$\begin{aligned} \boldsymbol{M} &= \boldsymbol{A} + \boldsymbol{N} \\ &= \boldsymbol{F}(\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi} + \boldsymbol{N} \\ &= (\boldsymbol{F} + \boldsymbol{N}^{(1)}, \boldsymbol{F}\boldsymbol{K} + \boldsymbol{N}^{(2)})\boldsymbol{\Pi} \end{aligned} \tag{2}$$

where $\boldsymbol{N}^{(1)}$ and $\boldsymbol{N}^{(2)}$ are d-by-r and d-by-ℓ submatrices of $\boldsymbol{N}$ such that $\boldsymbol{N}\boldsymbol{\Pi}^{-1} = (\boldsymbol{N}^{(1)}, \boldsymbol{N}^{(2)})$. Hereinafter, we use the notation ℓ to denote $m - r$. The goal of Problem 1 is to identify an index set $\mathcal{I}$ such that $\boldsymbol{M}(\mathcal{I}) = \boldsymbol{F} + \boldsymbol{N}^{(1)}$.

As mentioned in Section 1, it is ideal that an algorithm for Problem 1 has correctness and robustness properties. These properties are formally described as follows:

- **Correctness.** If the data matrix $\boldsymbol{M}$ does not contain a noise matrix $\boldsymbol{N}$ and is just a separable matrix, the algorithm returns an index set $\mathcal{I}$ such that $\boldsymbol{M}(\mathcal{I}) = \boldsymbol{F}$.

- **Robustness.** If the data matrix $\boldsymbol{M}$ contains a noise matrix $\boldsymbol{N}$ and is a noisy separable matrix such that $||\boldsymbol{N}||_p < \epsilon$, the algorithm returns an index set $\mathcal{I}$ such that $||\boldsymbol{M}(\mathcal{I}) - \boldsymbol{F}||_p < \tau\epsilon$ for some constant real number τ.

In particular, the robustness property has $\tau = 1$, if an algorithm can identify an index set $\mathcal{I}$ such that $\boldsymbol{M}(\mathcal{I}) = \boldsymbol{F}+\boldsymbol{N}^{(1)}$ where $\boldsymbol{F}$ and $\boldsymbol{N}^{(1)}$ are of (2) since $||\boldsymbol{M}(\mathcal{I})-\boldsymbol{F}||_p = ||\boldsymbol{N}^{(1)}||_p < \epsilon$.

In the design of the algorithm, some assumptions are usually placed on a separable matrix. Our algorithm uses Assumption 1.

Assumption 1 *A separable matrix $\boldsymbol{A}$ of (1) consists of an basis matrix $\boldsymbol{F}$ and a weight matrix $\boldsymbol{W}$ satisfying the following conditions.*

1-a) Every column vector of weight matrix $\boldsymbol{W}$ has unit 1-norm.

1-b) The basis matrix $\boldsymbol{F}$ has full column rank.

Assumption 1-a can be invoked without loss of generality. If the ith column of $\boldsymbol{W}$ is zero, so is the ith column of $\boldsymbol{A}$. Therefore, we can construct a smaller separable matrix having $\boldsymbol{W}$ with no zero column. Also, since we have

$$\boldsymbol{A} = \boldsymbol{F}\boldsymbol{W} \Leftrightarrow \boldsymbol{A}\boldsymbol{D} = \boldsymbol{F}\boldsymbol{W}\boldsymbol{D},$$

every column of $\boldsymbol{W}$ can have unit 1-norm. Here, $\boldsymbol{D}$ denotes a diagonal matrix having the (i,i)th diagonal element $d_{ii} = 1/||\boldsymbol{w}_i||_1$.

The same assumption is used by the algorithm in Gillis and Vavasis (2014). We may get the feeling that 1-b is strong. The algorithms (Arora et al., 2012a; Bittorf et al., 2012; Gillis, 2013; Gillis and Luce, 2013; Kumar et al., 2013) instead assume *simpliciality*, wherein no column vector of $\boldsymbol{F}$ can be represented as a convex hull of the remaining vectors of $\boldsymbol{F}$. Although 1-b is a stronger assumption, it still seems reasonable for Problem 1 from the standpoint of practical application. This is because, in such cases, it is less common for the column vectors of the basis matrix $\boldsymbol{F}$ to be linearly dependent.

2.2 Outline of Proposed Algorithm

Let us take a look at Problem 1 from a geometric point of view. For simplicity, consider the noiseless case first. Here, a data matrix is just a separable matrix $\boldsymbol{A}$. Separability implies that $\boldsymbol{A}$ has a factorization of the form (1). Under Assumption 1, $\mathrm{conv}(\boldsymbol{A})$ becomes an $(r-1)$-dimensional simplex in $\mathbb{R}^d$. The left part of Figure 1 visualizes a separable data matrix. The white points are data points, and the black ones are basis vectors. The key observation is that the basis vectors $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_r$ of $\boldsymbol{A}$ correspond to the vertices of $\mathrm{conv}(\boldsymbol{A})$. This is due to separability. Therefore, if all vertices of $\mathrm{conv}(\boldsymbol{A})$ can be found, we can obtain the basis matrix $\boldsymbol{F}$ of $\boldsymbol{A}$. This is not hard task, and we can design an efficient algorithm for doing it. But, if noise is added to a separable matrix, the task becomes hard. Let us suppose that the data matrix of Problem 1 is a noisy separable matrix $\widetilde{\boldsymbol{A}}$ of the form $\boldsymbol{A}+\boldsymbol{N}$. The vertices of $\mathrm{conv}(\widetilde{\boldsymbol{A}})$ do not necessarily match the basis vectors $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_r$ of $\boldsymbol{A}$. The right part of Figure 1 visualizes a noisy separable data matrix. This is the main reason why it is hard to identify the basis matrix from noisy separable one.

Our algorithm is designed on the basis of Proposition 3 in Section 4.1; it states that all vertices of a simplex can be found by using an ellipsoid. We here describe the proposition

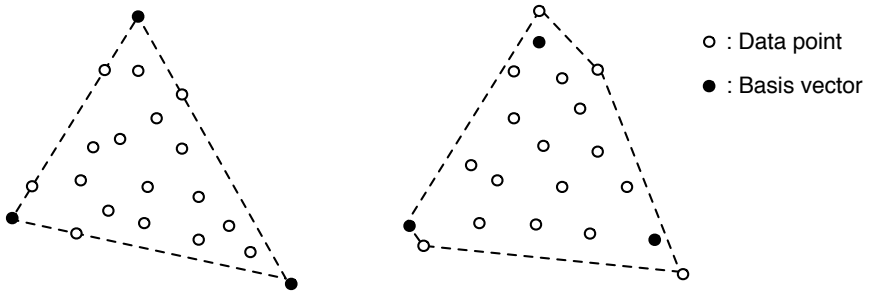

Figure 1: Convex hull of a separable data matrix with $r = 3$ under Assumption 1. (Left) Noiseless case. (Right) Noisy case.

from a geometric point of view. Consider an $(r-1)$-dimensional simplex Δ in $\mathbb{R}^r$. Let $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_r \in \mathbb{R}^r$ be the vertices of Δ, and $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell \in \mathbb{R}^r$ be the points in Δ. We draw the MVEE centered at the origin for a set $\mathcal{S} = \{\pm\boldsymbol{g}_1, \ldots, \pm\boldsymbol{g}_r, \pm\boldsymbol{b}_1, \ldots, \pm\boldsymbol{b}_\ell\}$. Then, the proposition says that the ellipsoid only touches the points $\pm\boldsymbol{g}_1, \ldots, \pm\boldsymbol{g}_r$ among all the points in $\mathcal{S}$. Therefore, the vertices of Δ can be found by checking whether the points in $\mathcal{S}$ lie on the boundary of ellipsoid. We should mention that the convex hull of the points in $\mathcal{S}$ becomes a full-dimensional crosspolytope in $\mathbb{R}^r$. Figure 2 illustrates the MVEE for a crosspolytope in $\mathbb{R}^3$.

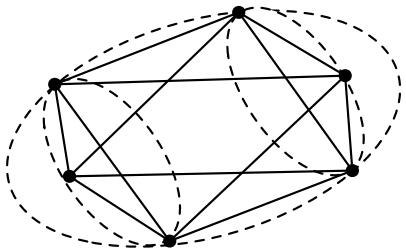

Figure 2: Minimum-volume enclosing ellipsoid for a full-dimensional crosspolytope in $\mathbb{R}^3$.

Under Assumption 1, the convex hull of a separable matrix $\boldsymbol{A}$ becomes an $(r-1)$-dimensional simplex in $\mathbb{R}^d$. Therefore, we rotate and embed the simplex in $\mathbb{R}^r$ by using an orthogonal transformation. Such a transformation can be obtained by singular value decomposition (SVD) of $\boldsymbol{A}$.

Now let us outline our algorithm for Problem 1. In this description, we assume for simplicity that the data matrix is a separable one $\boldsymbol{A} \in \mathbb{R}_+^{d\times m}$. First, the algorithm constructs an orthogonal transformation through the SVD of $\boldsymbol{A}$. By applying the transformation, it transforms $\boldsymbol{A}$ into a matrix $\boldsymbol{P} \in \mathbb{R}^{r\times m}$ such that the $\mathrm{conv}(\boldsymbol{P})$ is an $(r-1)$-dimensional sim-

plex in $\mathbb{R}^r$. Next, it draws the MVEE centered at the origin for a set $\mathcal{S} = \{\pm \boldsymbol{p}_1, \ldots, \pm \boldsymbol{p}_m\}$, where $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$ are the column vectors of $\boldsymbol{P}$, and outputs r points lying on the ellipsoid.

We call the algorithm *ellipsoidal rounding*, abbreviated as ER. The main computational costs of ER are in computing the SVD of $\boldsymbol{A}$ and the MVEE for $\mathcal{S}$. The MVEE computation can be formulated as a tractable convex optimization problem with m variables. A polynomial-time algorithm exists, and it is also known that a hybrid of the interior-point algorithm and cutting plane algorithm works efficiently in practice.

In later sections, we will see that ER algorithm works well even if noise is added. In particular, we show that ER correctly identifies the near-basis vectors of a noisy separable matrix if the noise is smaller than some level. We consider a situation in which the noise exceeds that level. In such a situation, the shape of crosspolytope formed by the data points is considerably perturbed by the noise, and it is possible that the MVEE touches many points. We thus need to select r points from the points on the boundary of the ellipsoid. In this paper, we perform existing algorithms such as SPA (Gillis and Vavasis, 2014) and XRAY (Kumar et al., 2013) to make the selection. Hence, ER works as a preprocessor which filters out basis vector candidates from data points and enhances the performance of existing algorithms.

2.3 Related Work

First, we will review the algorithms for NMF of general nonnegative matrices. There are an enormous number of studies. A commonly used approach is to formulate it as a nonconvex optimization problem and compute the local solution. Let $\boldsymbol{A}$ be a d-by-m nonnegative matrix, and consider an optimization problem with matrix variables $\boldsymbol{F} \in \mathbb{R}^{d \times r}$ and $\boldsymbol{W} \in \mathbb{R}^{r \times m}$,

$$\text{minimize } ||\boldsymbol{F}\boldsymbol{W} - \boldsymbol{A}||_F^2 \text{ subject to } \boldsymbol{F} \geq \boldsymbol{0} \text{ and } \boldsymbol{W} \geq \boldsymbol{0}.$$

This is an intractable nonconvex optimization problem, and in fact, it was shown to be NP-hard by Vavasis (2009). Therefore, the research target is in how to compute the local solution efficiently. It is popular to use the block coordinate descent (BCD) algorithm for this purpose. The algorithm solves the problem by alternately fixing the variables $\boldsymbol{F}$ and $\boldsymbol{W}$. The problem obtained by fixing either of $\boldsymbol{F}$ and $\boldsymbol{W}$ becomes a convex optimization problem. The existing studies propose to use, for instance, the projected gradient algorithm (Lin, 2007) and its variant (Lee and Seung, 2001), active set algorithm (Kim and Park, 2008, 2011), and projected quasi-Newton algorithm (Gong and Zhang, 2012). It is reported that the BCD algorithm shows good performance on average in computing NMFs. However, its performance depends on how we choose the initial point for starting the algorithm. We refer the reader to Kim et al. (2014) for a survey on the algorithms for NMF.

Next, we will survey the algorithms that work on noisy separable matrices. Four types of algorithm can be found:

- **AGKM (Arora et al., 2012a).** The algorithm constructs r sets of data points such that all of the basis vectors are contained in the union of the sets and each set has one basis vector. The construction entails solving m linear programming (LP) problems with $m - 1$ variables. Then, it chooses one element from each set, and outputs them.

- **Hottopixx (Bittorf et al., 2012; Gillis, 2013; Gillis and Luce, 2013).** Let $\boldsymbol{A}$ be a separable matrix of the form $\boldsymbol{F}(\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi}$. Consider a matrix $\boldsymbol{C}$ such that

$$\boldsymbol{C} = \boldsymbol{\Pi}^{-1} \begin{pmatrix} \boldsymbol{I} & \boldsymbol{K} \\ \boldsymbol{0} & \boldsymbol{0} \end{pmatrix} \boldsymbol{\Pi} \in \mathbb{R}^{m \times m}.$$

  It satisfies $\boldsymbol{A} = \boldsymbol{AC}$, and also, if the diagonal element is one, the position of its diagonal element indicates the index of basis vector in $\boldsymbol{A}$. The algorithm models $\boldsymbol{C}$ as the variable of an LP problem. It entails solving a single LP problem with m^2 variables.

- **SPA (Gillis and Vavasis, 2014).** Let $\boldsymbol{A}$ be a separable matrix of size d-by-m, and $\mathcal{S}$ be the set of the column vectors of $\boldsymbol{A}$. The algorithm is based on the following observation. Under Assumption 1, the maximum of a convex function over the elements in $\mathcal{S}$ is attained at the vertex of $\mathrm{conv}(\boldsymbol{A})$. The algorithm finds one element $\boldsymbol{a}$ in $\mathcal{S}$ that maximizes a convex function, and then, projects all elements in $\mathcal{S}$ into the orthogonal space to $\boldsymbol{a}$. This procedure is repeated until r elements are found.

- **XRAY (Kumar et al., 2013).** The algorithm has a similar spirit as SPA, but it uses a linear function instead of a convex one. Let $\boldsymbol{A}$ be a separable matrix of size d-by-m and $\mathcal{S}$ be the set of the column vectors of $\boldsymbol{A}$. Let $\mathcal{I}_k$ be the index set obtained after the kth iteration. This is a subset of $\{1, \ldots, m\}$ with cardinality k. In the $(k+1)$th iteration, it computes a residual matrix $\boldsymbol{R} = \boldsymbol{A}(\mathcal{I}_k)\boldsymbol{X}^* - \boldsymbol{A}$, where

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X} \geq \boldsymbol{0}} ||\boldsymbol{A}(\mathcal{I}_k)\boldsymbol{X} - \boldsymbol{A}||_2^2,$$

  and picks up one of the column vectors $\boldsymbol{r}_i$ of $\boldsymbol{R}$. Then, it finds one element from $\mathcal{S}$ which maximizes a linear function having $\boldsymbol{r}_i$ as the normal vector. Finally, $\mathcal{I}_k$ is updated by adding the index of the obtained element. This procedure is repeated until r indices are found. The performance of XRAY depends on how we select the column vector of the residual matrix $\boldsymbol{R}$ for making the linear function. Several ways of selection, called "rand", "max", "dist" and "greedy", have been proposed by the authors.

The next section describes the properties of these algorithms.

2.4 Comparison with Existing Algorithm

We compare the correctness and robustness properties of ER with those of AGKM, Hottopixx, SPA, and XRAY. ER is shown to have the two properties in Theorems 5 and 9. In particular, our robustness property in Theorem 9 states that ER correctly identifies the near-basis matrix of a noisy separable one $\widetilde{\boldsymbol{A}}$, and a robustness property with $\tau = 1$ holds if we set

$$\epsilon = \frac{\sigma(1-\mu)}{4} \tag{3}$$

and $p = 2$ under Assumption 1. Here, σ is the minimum singular value of the basis matrix $\boldsymbol{F}$ of a separable one $\boldsymbol{A}$ in the $\widetilde{\boldsymbol{A}}$, that is, $\sigma = \sigma_r(\boldsymbol{F})$, and μ is $\mu(\boldsymbol{K})$:

$$\mu(\boldsymbol{K}) = \max_{i=1,\ldots,\ell} ||\boldsymbol{k}_i||_2$$

for a weight matrix $\boldsymbol{K}$ of $\boldsymbol{A}$. Under Assumption 1-a, we have $\mu \leq 1$, and in particular, equality holds if and only if $\boldsymbol{k}_i$ has only one nonzero element.

All four of the existing algorithms have been shown to have a correctness property, whereas every one except XRAY has a robustness property. Hottopixx is the most similar to ER. Bittorf et al. (2012) showed that it has the correctness and robustness with $\tau = 1$ properties if one sets

$$\epsilon = \frac{\alpha \min\{d_0, \alpha\}}{9(r+1)} \tag{4}$$

and $p = 1$ under simpliciality and other assumptions. Here, α and d_0 are as follows. α is the minimum value of $\delta_{\boldsymbol{F}}(j)$ for $j = 1, \ldots, r$, where $\delta_{\boldsymbol{F}}(j)$ denotes an ℓ_1-distance between the jth column vector $\boldsymbol{f}_j$ of $\boldsymbol{F}$ and the convex hull of the remaining column vectors in $\boldsymbol{F}$. d_0 is the minimum value of $||\boldsymbol{a}_i - \boldsymbol{f}_j||_1$ for every i such that $\boldsymbol{a}_i$ is not a basis vector, and every $j = 1, \ldots, r$. The robustness of Hottopixx is further analyzed (Gillis, 2013; Gillis and Luce, 2013).

It can be interpreted that the ϵ of ER (3) is given by the multiplication of two parameters representing flatness and closeness of a given data matrix since σ measures the flatness of the convex hull of data points, and $1 - \mu$ measures the closeness between basis vectors and data points. Intuitively, we may say that an algorithm becomes sensitive to noise when a data matrix has the following features; one is that the convex hull of data points is close to a flat shape, and another is that there are data points close to basis vectors. The ϵ of (3) well matches the intuition. We see a similar structure in the ϵ of Hottopixx (4) since α and d_0 respectively measure the flatness and closeness of a given data.

Compared with Hottopixx, the ϵ of ER (3) does not contain $1/r$, and hence, it does not decrease as r increases. However, Assumption 1-b of ER is stronger than the simpliciality of Hottopixx. In a practical implementation, ER can handle a large matrix, while Hottopixx may have limitations on the size of the matrix it can handle. Hottopixx entails solving an LP problem with m^2 variables. In the NMFs arising in applications, m tends to be a large number. Although an LP is tractable, it becomes harder to solve as the size increases. Through experiments, we assessed the performance of Hottopixx with the CPLEX LP solver. The experiments showed that the algorithm had out of memory issues when m exceeded 2,000 with $d = 100$. Bittorf et al. (2012) proposed a parallel implementation to resolve these computational issues.

AGKM and SPA were shown to have a robustness property with $\tau \geq 1$ for some ϵ in Arora et al. (2012a) and Gillis and Vavasis (2014), respectively. In practical implementations, SPA and XRAY are scalable to the problem size and experimentally show good robustness. Section 6 reports a numerical comparison of ER with SPA and XRAY.

3. Review of Formulation and Algorithm for MVEE Computation

We review the formulation for computing the MVEE for a set of points, and survey the existing algorithms for the computation.

First of all, let us recall the terminology related to an ellipsoid. An ellipsoid in $\mathbb{R}^d$ is defined as a set $\mathcal{E}(\boldsymbol{L}, \boldsymbol{z}) = \{\boldsymbol{x} \in \mathbb{R}^d : (\boldsymbol{x} - \boldsymbol{z})^\top \boldsymbol{L}(\boldsymbol{x} - \boldsymbol{z}) \leq 1\}$ for a positive definite matrix $\boldsymbol{L}$ of size d and a vector $\boldsymbol{z} \in \mathbb{R}^d$. Here, $\boldsymbol{L}$ determines the shape of the ellipsoid and $\boldsymbol{z}$ is the center. Let $\boldsymbol{x}$ be a point in an ellipsoid $\mathcal{E}(\boldsymbol{L}, \boldsymbol{z})$. If the point $\boldsymbol{x}$ satisfies the equality

$(\boldsymbol{x}-\boldsymbol{z})^\top \boldsymbol{L}(\boldsymbol{x}-\boldsymbol{z})=1$, we call it an *active point* of the ellipsoid. In other words, an active point is one lying on the boundary of the ellipsoid.

The volume of the ellipsoid is given as $c(d)/\sqrt{\det \boldsymbol{L}}$, where $c(d)$ represents the volume of a unit ball in $\mathbb{R}^d$ and it is a real number depending on the dimension d. ER algorithm considers d-dimensional ellipsoids containing a set $\mathcal{S}$ of points in $\mathbb{R}^d$, and in particular, finds the minimum volume ellipsoid centered at the origin. In this paper, such an ellipsoid is referred to as an origin-centered MVEE for short.

Now, we are ready to describe a formulation for computing the origin-centered MVEE for a set of points. For m points $\boldsymbol{p}_1,\ldots,\boldsymbol{p}_m\in\mathbb{R}^d$, let $\mathcal{S}=\{\pm\boldsymbol{p}_1,\ldots,\pm\boldsymbol{p}_m\}$. The computation of the origin-centered MVEE for $\mathcal{S}$ is formulated as

$$\begin{array}{rlll}\mathbb{Q}(\mathcal{S}): & \text{minimize} & -\log\det \boldsymbol{L}, & \\ & \text{subject to} & \langle \boldsymbol{p}_i\boldsymbol{p}_i^\top, \boldsymbol{L}\rangle \le 1, & i=1,\ldots,m,\\ & & \boldsymbol{L}\succ \boldsymbol{0}, & \end{array}$$

where the matrix $\boldsymbol{L}$ of size d is the decision variable. The optimal solution $\boldsymbol{L}^*$ of $\mathbb{Q}$ gives the origin-centered MVEE for $\mathcal{S}$ as $\mathcal{E}(\boldsymbol{L}^*)=\{\boldsymbol{x}:\boldsymbol{x}^\top\boldsymbol{L}^*\boldsymbol{x}\le 1\}$. We here introduce some terminology. An active point of $\mathcal{E}(\boldsymbol{L}^*)$ is a vector $\boldsymbol{p}_i\in\mathbb{R}^d$ satisfying $\boldsymbol{p}_i^\top\boldsymbol{L}^*\boldsymbol{p}_i=1$. We call $\boldsymbol{p}_i$ an *active point of* $\mathbb{Q}(\mathcal{S})$, and the index i of $\boldsymbol{p}_i$ an *active index of* $\mathbb{Q}(\mathcal{S})$. The ellipsoid $\mathcal{E}(\boldsymbol{L}^*)$ is centrally symmetric, and if a vector $\boldsymbol{p}_i$ is an active point, so is $-\boldsymbol{p}_i$. The dual of $\mathbb{Q}$ reads

$$\begin{array}{rlll}\mathbb{Q}_*(\mathcal{S}): & \text{maximize} & \log\det \Omega(\boldsymbol{u}), & \\ & \text{subject to} & \boldsymbol{e}^\top\boldsymbol{u}=1, & \\ & & \boldsymbol{u}\ge \boldsymbol{0}, & \end{array}$$

where the vector $\boldsymbol{u}$ is the decision variable. Here, $\Omega:\mathbb{R}^m\to\mathbb{S}^d$ is a linear function given as $\Omega(\boldsymbol{u})=\sum_{i=1}^m \boldsymbol{p}_i\boldsymbol{p}_i^\top u_i$; equivalently, $\Omega(\boldsymbol{u})=\boldsymbol{P}\mathrm{diag}(\boldsymbol{u})\boldsymbol{P}^\top$ for $\boldsymbol{P}=(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_m)\in\mathbb{R}^{d\times m}$. It follows from the Karush-Kuhn-Tucker (KKT) conditions for these problems that the optimal solution $\boldsymbol{L}^*$ of $\mathbb{Q}$ is represented by $\frac{1}{d}\Omega(\boldsymbol{u}^*)^{-1}$ for the optimal solution $\boldsymbol{u}^*$ of $\mathbb{Q}_*$. We make the following assumption to ensure the existence of an optimal solution of $\mathbb{Q}$.

Assumption 2 *rank*$(\boldsymbol{P})=d$ *for* $\boldsymbol{P}=(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_m)\in\mathbb{R}^{d\times m}$.

Later, the KKT conditions will play an important role in our discussion of the active points of $\mathbb{Q}$. Here though, we will describe the conditions: $\boldsymbol{L}^*\in\mathbb{S}^d$ is an optimal solution for $\mathbb{Q}$ and $\boldsymbol{z}^*\in\mathbb{R}^m$ is the associated Lagrange multiplier vector if and only if there exist $\boldsymbol{L}^*\in\mathbb{S}^d$ and $\boldsymbol{z}^*\in\mathbb{R}^m$ such that

$$-(\boldsymbol{L}^*)^{-1}+\Omega(\boldsymbol{z}^*)=\boldsymbol{0}, \tag{5}$$
$$z_i^*(\langle \boldsymbol{p}_i\boldsymbol{p}_i^\top,\boldsymbol{L}^*\rangle-1)=0,\quad i=1,\ldots,m, \tag{6}$$
$$\langle \boldsymbol{p}_i\boldsymbol{p}_i^\top,\boldsymbol{L}^*\rangle\le 1,\quad i=1,\ldots,m, \tag{7}$$
$$\boldsymbol{L}^*\succ\boldsymbol{0}, \tag{8}$$
$$z_i^*\ge 0,\quad i=1,\ldots,m. \tag{9}$$

Many algorithms have been proposed for solving problems $\mathbb{Q}$ and $\mathbb{Q}_*$. These can be categorized into mainly two types: conditional gradient algorithms (also referred to as

Frank-Wolfe algorithms) and interior-point algorithms. Below, we survey the studies on these two algorithms.

Khachiyan (1996) proposed a barycentric coordinate descent algorithm, which can be interpreted as a conditional gradient algorithm. He showed that the algorithm has a polynomial-time iteration complexity. Several researchers investigated and revised Khachiyan's algorithm. Kumar and Yildirim (2005) showed that the iteration complexity of Khachiyan's algorithm can be slightly reduced if it starts from a well-selected initial point. Todd and Yildirim (2007) and Ahipasaoglu et al. (2008) incorporated a step called as a Wolfe's away-step. The revised algorithm was shown to have a polynomial-time iteration complexity and a linear convergence rate.

A dual interior-point algorithm was given by Vandenberghe et al. (1998). A primal-dual interior-point algorithm was given by Toh (1999), and numerical experiments showed that this algorithm is efficient and can provide accurate solutions. A practical algorithm was designed by Sun and Freund (2004) for solving large-scale problems. In particular, a hybrid of the interior-point algorithm and cutting plane algorithm was shown to be efficient in numerical experiments. For instance, the paper reported that the hybrid algorithm can solve problems with $d = 30$ and $m = 30,000$ in under 30 seconds on a personal computer. Tsuchiya and Xia (2007) considered generalized forms of $\mathbb{Q}$ and $\mathbb{Q}_*$ and showed that a primal-dual interior-point algorithm for the generalized forms has a polynomial-time iteration complexity.

Next, let us discuss the complexity of these two sorts of algorithms for $\mathbb{Q}$ and $\mathbb{Q}_*$. In each iteration, the arithmetic operations of the conditional gradient algorithms are less than those of the interior-point algorithms. Each iteration of a conditional gradient algorithm (Khachiyan, 1996; Kumar and Yildirim, 2005; Todd and Yildirim, 2007; Ahipasaoglu et al., 2008) requires $O(md)$ arithmetic operations. On the other hand, assuming that the number of data points m is sufficiently larger than the dimension of data points d, the main complexity of interior-point algorithms (Vandenberghe et al., 1998; Toh, 1999) comes from solving an m-by-m system of linear equations in each iteration. The solution serves as the search direction for the next iteration. Solving these linear equations requires $O(m^3)$ arithmetic operations. In practice, the number of iterations of conditional gradient algorithms is much larger than that of interior-point algorithms. Ahipasaoglu et al. (2008) reports that conditional gradient algorithms take several thousands iterations to solve problems such that d runs from 10 to 30 and m from $10,000$ to $30,000$. On the other hand, Sun and Freund (2004) reports that interior-point algorithms usually terminate after several dozen iterations and provide accurate solutions.

One of the concerns about interior-point algorithms is the computational cost of each iteration. It is possible to reduce the cost considerably by using a cutting plane strategy. A hybrid of interior-point algorithm and cutting plane algorithm has an advantage over conditional gradient algorithms. In fact, Ahipasaoglu et al. (2008) reports that the hybrid algorithm is faster than the conditional gradient algorithms and works well even on large problems. Therefore, we use the hybrid algorithm to solve $\mathbb{Q}$ in our practical implementation of ER. The details are in Section 5.1.

Here, it should be mentioned that this paper uses a terminology "cutting plane strategy" for what other papers (Sun and Freund, 2004; Ahipasaoglu et al., 2008) have called the

"active set strategy", since it might be confused with "active set algorithm" for solving a nonnegative least square problem.

4. Description and Analysis of the Algorithm

The ER algorithm is presented below. Throughout of this paper, we use the notation $\mathbb{N}$ to denote a set of nonnegative integer numbers.

Algorithm 1 Ellipsoidal Rounding (ER) for Problem 1

Input: $\boldsymbol{M} \in \mathbb{R}_+^{d\times m}$ and $r \in \mathbb{N}$.
Output: $\mathcal{I}$.

1: Compute the SVD of $\boldsymbol{M}$, and construct the reduced matrix $\boldsymbol{P} \in \mathbb{R}^{r\times m}$ associated with r.

2: Let $\mathcal{S} = \{\pm\boldsymbol{p}_1, \ldots, \pm\boldsymbol{p}_m\}$ for the column vectors $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$ of $\boldsymbol{P}$. Solve $\mathbb{Q}(\mathcal{S})$, and construct the active index set $\mathcal{I}$.

Step 1 needs to be explained in detail. Let $\boldsymbol{M}$ be a noisy separable matrix of size d-by-m. In general, the $\boldsymbol{M}$ is a full-rank due to the existence of a noise matrix. However, the rank is close to r when the amount of noise is small, and in particular, it is r in the noiseless case. Accordingly, we construct a low-rank approximation matrix to $\boldsymbol{M}$ and reduce the redundancy in the space spanned by the column vectors of $\boldsymbol{M}$.

We use an SVD for the construction of the low-rank approximation matrix. The SVD of $\boldsymbol{M}$ gives a decomposition of the form,

$$\boldsymbol{M} = \boldsymbol{U}\Sigma\boldsymbol{V}^\top.$$

Here, $\boldsymbol{U}$ and $\boldsymbol{V}$ are d-by-d and m-by-m orthogonal matrices, respectively. In this paper, we call the $\boldsymbol{U}$ a *left orthogonal matrix* of the SVD of $\boldsymbol{M}$. Let $t = \min\{d, m\}$. Σ is a rectangular diagonal matrix consisting of the singular values $\sigma_1, \ldots, \sigma_t$ of $\boldsymbol{M}$, and it is of the form,

$$\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_t) \in \mathbb{R}^{d\times m}$$

with $\sigma_1 \geq \cdots \geq \sigma_t \geq 0$. By choosing the top r singular values while setting the others to 0 in Σ, we construct

$$\Sigma^r = \mathrm{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0) \in \mathbb{R}^{d\times m}$$

and let

$$\boldsymbol{M}^r = \boldsymbol{U}\Sigma^r\boldsymbol{V}^\top.$$

$\boldsymbol{M}^r$ is the best rank-r approximation to $\boldsymbol{M}$ as measured by the matrix 2-norm and satisfies $||\boldsymbol{M} - \boldsymbol{M}^r||_2 = \sigma_{r+1}$ (see, for instance, Theorem 2.5.3 of Golub and Loan, 1996). By applying the left orthogonal matrix $\boldsymbol{U}^\top$ to $\boldsymbol{M}^r$, we have

$$\boldsymbol{U}^\top\boldsymbol{M}^r = \begin{pmatrix} \boldsymbol{P} \\ \boldsymbol{0} \end{pmatrix} \in \mathbb{R}^{d\times m},$$

where $\boldsymbol{P}$ is an r-by-m matrix with $\mathrm{rank}(\boldsymbol{P}) = r$. We call such a matrix $\boldsymbol{P}$ a *reduced matrix of $\boldsymbol{M}$ associated with r*. Since Assumption 2 holds for the $\boldsymbol{P}$, it is possible to perform an MVEE computation for a set of the column vectors.

4.1 Correctness for a Separable Matrix

We analyze the correctness property of Algorithm 1. Let $\boldsymbol{A}$ be a separable matrix of size d-by-m. Assume that Assumption 1 holds for $\boldsymbol{A}$. We run Algorithm 1 for $(\boldsymbol{A}, \mathrm{rank}(\boldsymbol{A}))$. Step 1 computes the reduced matrix $\boldsymbol{P}$ of $\boldsymbol{A}$. Since $r = \mathrm{rank}(\boldsymbol{A})$, we have $\boldsymbol{A} = \boldsymbol{A}^r$, where $\boldsymbol{A}^r$ is the best rank-r approximation matrix to $\boldsymbol{A}$. Let $\boldsymbol{U} \in \mathbb{R}^{d\times d}$ be the left orthogonal matrix of the SVD of $\boldsymbol{A}$. The reduced matrix $\boldsymbol{P} \in \mathbb{R}^{r\times m}$ of $\boldsymbol{A}$ is obtained as

$$\begin{aligned} \begin{pmatrix} \boldsymbol{P} \\ \boldsymbol{0} \end{pmatrix} &= \boldsymbol{U}^\top \boldsymbol{A} \\ &= \boldsymbol{U}^\top \boldsymbol{F}(\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi}. \end{aligned} \tag{10}$$

From the above, we see that

$$\boldsymbol{U}^\top \boldsymbol{F} = \begin{pmatrix} \boldsymbol{G} \\ \boldsymbol{0} \end{pmatrix} \in \mathbb{R}^{d\times m}, \text{ where } \boldsymbol{G} \in \mathbb{R}^{r\times r}. \tag{11}$$

Here, we have $\mathrm{rank}(\boldsymbol{G}) = r$ since $\mathrm{rank}(\boldsymbol{F}) = r$ by Assumption 1-b and $\boldsymbol{U}$ is an orthogonal matrix. By using $\boldsymbol{G}$, we rewrite $\boldsymbol{P}$ as

$$\boldsymbol{P} = (\boldsymbol{G}, \boldsymbol{G}\boldsymbol{K})\boldsymbol{\Pi}.$$

From Assumption 1-a, the column vectors $\boldsymbol{k}_i$ of the weight matrix $\boldsymbol{K} \in \mathbb{R}^{r\times \ell}$ satisfy the conditions

$$||\boldsymbol{k}_i||_1 = 1 \text{ and } \boldsymbol{k}_i \geq \boldsymbol{0}, \quad i = 1, \ldots, \ell. \tag{12}$$

In Step 2, we collect the column vectors of $\boldsymbol{P}$ and construct a set $\mathcal{S}$ of them. Let $\boldsymbol{B} = \boldsymbol{G}\boldsymbol{K}$, and let $\boldsymbol{g}_j$ and $\boldsymbol{b}_i$ be the column vector of $\boldsymbol{G}$ and $\boldsymbol{B}$, respectively. $\mathcal{S}$ is a set of vectors $\pm\boldsymbol{g}_1, \ldots, \pm\boldsymbol{g}_r, \pm\boldsymbol{b}_1, \ldots, \pm\boldsymbol{b}_\ell$. The following proposition guarantees that the active points of $\mathbb{Q}(\mathcal{S})$ are $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_r$. We can see from (10) and (11) that the index set of the column vectors of $\boldsymbol{G}$ is identical to that of of $\boldsymbol{F}$. Hence, the basis matrix $\boldsymbol{F}$ of a separable one $\boldsymbol{A}$ can be obtained by finding the active points of $\mathbb{Q}(\mathcal{S})$.

Proposition 3 *Let $\boldsymbol{G} \in \mathbb{R}^{r\times r}$ and $\boldsymbol{B} = \boldsymbol{G}\boldsymbol{K} \in \mathbb{R}^{r\times \ell}$ for $\boldsymbol{K} \in \mathbb{R}^{r\times \ell}$. For the column vectors $\boldsymbol{g}_j$ and $\boldsymbol{b}_i$ of $\boldsymbol{G}$ and $\boldsymbol{B}$, respectively, let $\mathcal{S} = \{\pm\boldsymbol{g}_1, \ldots \pm \boldsymbol{g}_r, \pm\boldsymbol{b}_1, \ldots, \pm\boldsymbol{b}_\ell\}$. Suppose that rank$(\boldsymbol{G}) = r$ and $\boldsymbol{K}$ satisfies the condition (12). Then, the active point set of $\mathbb{Q}(\mathcal{S})$ is $\{\boldsymbol{g}_1, \ldots, \boldsymbol{g}_r\}$.*

Proof We show that an optimal solution $\boldsymbol{L}^*$ of $\mathbb{Q}(\mathcal{S})$ is $(\boldsymbol{G}\boldsymbol{G}^\top)^{-1}$ and its associated Lagrange multiplier $\boldsymbol{z}^*$ is $(\boldsymbol{e}; \boldsymbol{0})$, where $\boldsymbol{e}$ is an r-dimensional all-ones vector and $\boldsymbol{0}$ is an ℓ-dimensional zero vector. Here, the Lagrange multipliers are one for the constraints $\langle \boldsymbol{g}_j\boldsymbol{g}_j^\top, \boldsymbol{L}\rangle \leq 1$, and these are zero for $\langle \boldsymbol{b}_i\boldsymbol{b}_i^\top, \boldsymbol{L}\rangle \leq 1$.

Since $\boldsymbol{G}$ is nonsingular, the inverse of $\boldsymbol{G}\boldsymbol{G}^\top$ exists and it is positive definite. Now we check that $\boldsymbol{L}^* = (\boldsymbol{G}\boldsymbol{G}^\top)^{-1}$ and $\boldsymbol{z}^* = (\boldsymbol{e}; \boldsymbol{0})$ satisfy the KKT conditions (5)-(9) for the problem. It was already seen that the conditions (5), (8), and (9) are satisfied. For the remaining conditions, we have

$$\langle \boldsymbol{g}_j\boldsymbol{g}_j^\top, (\boldsymbol{G}\boldsymbol{G}^\top)^{-1}\rangle = (\boldsymbol{G}^\top(\boldsymbol{G}\boldsymbol{G}^\top)^{-1}\boldsymbol{G})_{jj} = 1 \tag{13}$$

and

$$\begin{aligned}\langle \boldsymbol{b}_i \boldsymbol{b}_i^\top, (\boldsymbol{G}\boldsymbol{G}^\top)^{-1} \rangle &= (\boldsymbol{B}^\top (\boldsymbol{G}\boldsymbol{G}^\top)^{-1} \boldsymbol{B})_{ii} \\ &= (\boldsymbol{K}^\top \boldsymbol{G}^\top (\boldsymbol{G}\boldsymbol{G}^\top)^{-1} \boldsymbol{G}\boldsymbol{K})_{ii} \\ &= \boldsymbol{k}_i^\top \boldsymbol{k}_i \\ &\leq ||\boldsymbol{k}_i||_1^2 = 1. \end{aligned} \tag{14}$$

Here, $(\cdot)_{ii}$ for a matrix denotes the (i,i)th element of the matrix. The inequality in (14) follows from condition (12). Also, the Lagrange multipliers are zero for the inequality constraints $\langle \boldsymbol{b}_i \boldsymbol{b}_i^\top, (\boldsymbol{G}\boldsymbol{G}^\top)^{-1} \rangle \leq 1$. Thus, conditions (6) and (7) are satisfied. Accordingly, $(\boldsymbol{G}\boldsymbol{G}^\top)^{-1}$ is an optimal solution of $\mathbb{Q}(\mathcal{S})$.

We can see from (13) that $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_r$ are the active points of the problem. Moreover, we may have equality in (14). In fact, equality holds if and only if $\boldsymbol{k}_i$ has only one nonzero element. For such $\boldsymbol{k}_i$, $\boldsymbol{b}_i = \boldsymbol{G}\boldsymbol{k}_i$ coincides with some vector in $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_r$. ■

From the above discussion, we can immediately notice that this proposition holds if for a matrix $\boldsymbol{K} \in \mathbb{R}^{r \times \ell}$, the column vectors $\boldsymbol{k}_i$ satisfy

$$||\boldsymbol{k}_i||_2 < 1, \quad i = 1, \ldots, m. \tag{15}$$

Note that in contrast with condition (12), this condition does not require the matrix to be nonnegative.

Corollary 4 *Proposition 3 holds even if we suppose that $\boldsymbol{K} \in \mathbb{R}^{r \times \ell}$ satisfies condition (15), instead of condition (12).*

Note that this corollary is used to show the robustness of Algorithm 1 on a noisy separable matrix. The correctness of Algorithm 1 for a separable matrix follows from the above discussion and Proposition 3.

Theorem 5 *Let $\boldsymbol{A}$ be a separable matrix. Assume that Assumption 1 holds for $\boldsymbol{A}$. Then, Algorithm 1 for $(\boldsymbol{A}, rank(\boldsymbol{A}))$ returns an index set $\mathcal{I}$ such that $\boldsymbol{A}(\mathcal{I}) = \boldsymbol{F}$.*

4.2 Robustness for a Noisy Separable Matrix

Next, we analyze the robustness property of Algorithm 1. Let $\boldsymbol{A}$ be a separable matrix of size d-by-m. Assume that Assumption 1 holds for $\boldsymbol{A}$. Let $\widetilde{\boldsymbol{A}}$ be a noisy separable matrix of the form $\boldsymbol{A} + \boldsymbol{N}$. We run Algorithm 1 for $(\widetilde{\boldsymbol{A}}, \mathrm{rank}(\boldsymbol{A}))$. Step 1 computes the reduced matrix $\boldsymbol{P}$ of $\widetilde{\boldsymbol{A}}$. Let $\boldsymbol{U} \in \mathbb{R}^{d \times d}$ be the left orthogonal matrix of the SVD of $\widetilde{\boldsymbol{A}}$, and $\widetilde{\boldsymbol{A}}^r$ be the best rank-r approximation matrix to $\widetilde{\boldsymbol{A}}$. We denote the residual matrix $\widetilde{\boldsymbol{A}} - \widetilde{\boldsymbol{A}}^r$ by $\widetilde{\boldsymbol{A}}^r_\diamond$.

For the reduced matrix $\boldsymbol{P}$ of $\widetilde{\boldsymbol{A}}$, we have

$$\begin{aligned}
\begin{pmatrix} \boldsymbol{P} \\ \boldsymbol{0} \end{pmatrix} &= \boldsymbol{U}^\top \widetilde{\boldsymbol{A}}^r \\
&= \boldsymbol{U}^\top (\widetilde{\boldsymbol{A}} - \widetilde{\boldsymbol{A}}^r_\diamond) && (16) \\
&= \boldsymbol{U}^\top (\boldsymbol{A} + \boldsymbol{N} - \widetilde{\boldsymbol{A}}^r_\diamond) \\
&= \boldsymbol{U}^\top (\boldsymbol{A} + \bar{\boldsymbol{N}}) && (17) \\
&= \boldsymbol{U}^\top ((\boldsymbol{F}, \boldsymbol{F}\boldsymbol{K})\boldsymbol{\Pi} + \bar{\boldsymbol{N}}) \\
&= \boldsymbol{U}^\top (\boldsymbol{F} + \bar{\boldsymbol{N}}^{(1)}, \boldsymbol{F}\boldsymbol{K} + \bar{\boldsymbol{N}}^{(2)})\boldsymbol{\Pi} && (18) \\
&= \boldsymbol{U}^\top (\widehat{\boldsymbol{F}}, \widehat{\boldsymbol{F}}\boldsymbol{K} + \widehat{\boldsymbol{N}})\boldsymbol{\Pi}. && (19)
\end{aligned}$$

The following notation is used in the above: $\bar{\boldsymbol{N}} = \boldsymbol{N} - \widetilde{\boldsymbol{A}}^r_\diamond$ in (17); $\bar{\boldsymbol{N}}^{(1)}$ and $\bar{\boldsymbol{N}}^{(2)}$ in (18) are the d-by-r and d-by-ℓ submatrices of $\bar{\boldsymbol{N}}$ such that $\bar{\boldsymbol{N}}\boldsymbol{\Pi}^{-1} = (\bar{\boldsymbol{N}}^{(1)}, \bar{\boldsymbol{N}}^{(2)})$; $\widehat{\boldsymbol{F}} = \boldsymbol{F} + \bar{\boldsymbol{N}}^{(1)}$ and $\widehat{\boldsymbol{N}} = -\bar{\boldsymbol{N}}^{(1)}\boldsymbol{K} + \bar{\boldsymbol{N}}^{(2)}$ in (19). This implies that

$$\boldsymbol{U}^\top \widehat{\boldsymbol{F}} = \begin{pmatrix} \widehat{\boldsymbol{G}} \\ \boldsymbol{0} \end{pmatrix}, \text{ where } \widehat{\boldsymbol{G}} \in \mathbb{R}^{r \times r}, \tag{20}$$

and

$$\boldsymbol{U}^\top \widehat{\boldsymbol{N}} = \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}, \text{ where } \boldsymbol{R} \in \mathbb{R}^{r \times \ell}. \tag{21}$$

Hence, we can rewrite $\boldsymbol{P}$ as

$$\boldsymbol{P} = (\widehat{\boldsymbol{G}}, \widehat{\boldsymbol{G}}\boldsymbol{K} + \boldsymbol{R})\boldsymbol{\Pi}.$$

$\widetilde{\boldsymbol{A}}$ is represented by (2) as

$$\widetilde{\boldsymbol{A}} = (\widetilde{\boldsymbol{F}}, \widetilde{\boldsymbol{F}}\boldsymbol{K} + \widetilde{\boldsymbol{N}})\boldsymbol{\Pi},$$

where $\widetilde{\boldsymbol{F}}$ and $\widetilde{\boldsymbol{N}}$ denote $\boldsymbol{F} + \boldsymbol{N}^{(1)}$ and $-\boldsymbol{N}^{(1)}\boldsymbol{K} + \boldsymbol{N}^{(2)}$, respectively. From (16), we have

$$\begin{pmatrix} (\widehat{\boldsymbol{G}}, \widehat{\boldsymbol{G}}\boldsymbol{K} + \boldsymbol{R})\boldsymbol{\Pi} \\ \boldsymbol{0} \end{pmatrix} = \boldsymbol{U}^\top ((\widetilde{\boldsymbol{F}}, \widetilde{\boldsymbol{F}}\boldsymbol{K} + \widetilde{\boldsymbol{N}})\boldsymbol{\Pi} - \widetilde{\boldsymbol{A}}^r_\diamond).$$

Therefore, the index set of the column vectors of $\widehat{\boldsymbol{G}}$ is identical to that of $\widetilde{\boldsymbol{F}}$. If all the column vectors of $\widehat{\boldsymbol{G}}$ are found in $\boldsymbol{P}$, we can identify $\widetilde{\boldsymbol{F}}$ hidden in $\widetilde{\boldsymbol{A}}$.

In Step 2, we collect the column vectors of $\boldsymbol{P}$ and construct a set $\mathcal{S}$ of them. Let

$$\widehat{\boldsymbol{B}} = \widehat{\boldsymbol{G}}\boldsymbol{K} + \boldsymbol{R}, \tag{22}$$

and let $\widehat{\boldsymbol{g}}_j$ and $\widehat{\boldsymbol{b}}_i$ respectively be the column vectors of $\widehat{\boldsymbol{G}}$ and $\widehat{\boldsymbol{B}}$. $\mathcal{S}$ is a set of vectors $\pm\widehat{\boldsymbol{g}}_1, \ldots, \pm\widehat{\boldsymbol{g}}_r, \pm\widehat{\boldsymbol{b}}_1, \ldots, \pm\widehat{\boldsymbol{b}}_\ell$. We can see from Corollary 4 that, if $\mathrm{rank}(\widehat{\boldsymbol{G}}) = r$ and $\widehat{\boldsymbol{b}}_i$ is written as $\widehat{\boldsymbol{b}}_i = \widehat{\boldsymbol{G}}\widehat{\boldsymbol{k}}_i$ by using $\widehat{\boldsymbol{k}}_i \in \mathbb{R}^r$ with $||\widehat{\boldsymbol{k}}_i||_2 < 1$, the active points of $\mathbb{Q}(\mathcal{S})$ are given as the column vectors $\widehat{\boldsymbol{g}}_1, \ldots, \widehat{\boldsymbol{g}}_r$ of $\widehat{\boldsymbol{G}}$. Below, we examine the amount of noise $\boldsymbol{N}$ such that the conditions of Corollary 4 still hold.

Lemma 6 *Let $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{N} \in \mathbb{R}^{d \times m}$. Then, $|\sigma_i(\widetilde{\boldsymbol{A}}) - \sigma_i(\boldsymbol{A})| \leq ||\boldsymbol{N}||_2$ for each $i = 1, \ldots, t$ where $t = \min\{d, m\}$.*

Proof See Corollary 8.6.2 of Golub and Loan (1996). ■

Lemma 7 *Let* $n = ||\boldsymbol{N}||_2$ *and* $\mu = \mu(\boldsymbol{K})$.

7-a) The matrix $\bar{\boldsymbol{N}}$ *of (17) satisfies* $||\bar{\boldsymbol{N}}||_2 \leq 2n$.

7-b) The column vectors $\boldsymbol{r}_i$ *of matrix* $\boldsymbol{R}$ *of (21) satisfy* $||\boldsymbol{r}_i||_2 \leq 2n(\mu+1)$ *for* $i = 1, \ldots, m$.

7-c) The singular values of matrix $\widehat{\boldsymbol{G}}$ *of (20) satisfy* $|\sigma_i(\widehat{\boldsymbol{G}}) - \sigma_i(\boldsymbol{F})| \leq 2n$ *for* $i = 1, \ldots, r$.

Proof 7-a) Since $\bar{\boldsymbol{N}} = \boldsymbol{N} - \widetilde{\boldsymbol{A}}^r_\diamond$,

$$||\bar{\boldsymbol{N}}||_2 \leq ||\boldsymbol{N}||_2 + ||\widetilde{\boldsymbol{A}}^r_\diamond||_2.$$

We have $||\widetilde{\boldsymbol{A}}^r_\diamond||_2 \leq n$ since $||\widetilde{\boldsymbol{A}}^r_\diamond||_2 = \sigma_{r+1}(\widetilde{\boldsymbol{A}})$ and from Lemma 6, $|\sigma_{r+1}(\widetilde{\boldsymbol{A}}) - \sigma_{r+1}(\boldsymbol{A})| \leq n$. Therefore, $||\bar{\boldsymbol{N}}||_2 \leq 2n$.

7-b) Let $\widehat{\boldsymbol{n}}_i$ be the column vector of the matrix $\widehat{\boldsymbol{N}}$ of (19). Since $\boldsymbol{U}^\top \widehat{\boldsymbol{n}}_i = (\boldsymbol{r}_i; \boldsymbol{0})$ for an orthogonal matrix $\boldsymbol{U}$, we have $||\widehat{\boldsymbol{n}}_i||_2 = ||\boldsymbol{r}_i||_2$. Therefore, we will evaluate $||\widehat{\boldsymbol{n}}_i||_2$. Let $\boldsymbol{k}_i$ and $\bar{\boldsymbol{n}}^{(2)}_i$ be the column vectors of $\boldsymbol{K}$ and $\bar{\boldsymbol{N}}^{(2)}$, respectively. Then, $\widehat{\boldsymbol{n}}_i$ can be represented as $-\bar{\boldsymbol{N}}^{(1)}\boldsymbol{k}_i + \bar{\boldsymbol{n}}^{(2)}_i$. Thus, by Lemma 7-a, we have

$$||\boldsymbol{r}_i||_2 = ||\widehat{\boldsymbol{n}}_i||_2 \leq ||\bar{\boldsymbol{N}}^{(1)}||_2||\boldsymbol{k}_i||_2 + ||\bar{\boldsymbol{n}}^{(2)}_i||_2 \leq 2n(\mu+1).$$

7-c) Since $\boldsymbol{U}^\top \widehat{\boldsymbol{F}} = (\widehat{\boldsymbol{G}}; \boldsymbol{0})$ for an orthogonal matrix $\boldsymbol{U}$, the singular values of $\widehat{\boldsymbol{F}}$ and $\widehat{\boldsymbol{G}}$ are identical. Also, since $\widehat{\boldsymbol{F}} = \boldsymbol{F} + \bar{\boldsymbol{N}}^{(1)}$ and Lemma 6, we have

$$|\sigma_i(\widehat{\boldsymbol{G}}) - \sigma_i(\boldsymbol{F})| = |\sigma_i(\widehat{\boldsymbol{F}}) - \sigma_i(\boldsymbol{F})| \leq ||\bar{\boldsymbol{N}}^{(1)}||_2 \leq 2n.$$

■

The following lemma ensures that the conditions of Corollary 4 hold if the amount of noise is smaller than a certain level.

Lemma 8 *Let* $\widehat{\boldsymbol{G}}$ *be the matrix of (20), and let* $\widehat{\boldsymbol{b}}_i$ *be the column vector of* $\widehat{\boldsymbol{B}}$ *of (22). Suppose that* $||\boldsymbol{N}||_2 < \epsilon$ *for* $\epsilon = \frac{1}{4}\sigma(1-\mu)$ *where* $\sigma = \sigma_r(\boldsymbol{F})$ *and* $\mu = \mu(\boldsymbol{K})$. *Then,*

8-a) $rank(\widehat{\boldsymbol{G}}) = r$.

8-b) $\widehat{\boldsymbol{b}}_i$ *is represented as* $\widehat{\boldsymbol{G}}\widehat{\boldsymbol{k}}_i = \widehat{\boldsymbol{b}}_i$ *by using* $\widehat{\boldsymbol{k}}_i$ *such that* $||\widehat{\boldsymbol{k}}_i||_2 < 1$.

In the proof below, n denotes $||\boldsymbol{N}||_2$.

Proof 8-a) From Lemma 7-c, the minimum singular value of $\widehat{\boldsymbol{G}}$ satisfies

$$\begin{aligned} \sigma_r(\widehat{\boldsymbol{G}}) &\geq \sigma - 2n \\ &> \sigma - 2\epsilon = \frac{1}{2}\sigma(1+\mu) > 0. \end{aligned}$$

The final inequality follows from $\sigma > 0$ due to Assumption 1-b. Hence, we have $\mathrm{rank}(\widehat{\boldsymbol{G}}) = r$.

8-b) Let $\boldsymbol{k}_i$ and $\boldsymbol{r}_i$ be the column vectors of $\boldsymbol{K}$ and $\boldsymbol{R}$, respectively. Then, we have $\widehat{\boldsymbol{b}}_i = \widehat{\boldsymbol{G}}\boldsymbol{k}_i + \boldsymbol{r}_i$. Since Lemma 8-a guarantees that $\widehat{\boldsymbol{G}}$ has an inverse, it can be represented as $\widehat{\boldsymbol{b}}_i = \widehat{\boldsymbol{G}}\widehat{\boldsymbol{k}}_i$ by $\widehat{\boldsymbol{k}}_i = \boldsymbol{k}_i + \widehat{\boldsymbol{G}}^{-1}\boldsymbol{r}_i$. It follows from Lemmas 7-b and 7-c that

$$\begin{aligned} ||\widehat{\boldsymbol{k}}_i||_2 &\leq ||\boldsymbol{k}_i||_2 + ||\widehat{\boldsymbol{G}}^{-1}||_2||\boldsymbol{r}_i||_2 \\ &\leq \mu + \frac{2n(\mu+1)}{\sigma - 2n}. \end{aligned}$$

Since $n < \frac{1}{4}\sigma(1-\mu)$, we have $||\widehat{\boldsymbol{k}}_i||_2 < 1$. ■

The robustness of Algorithm 1 for a noisy separable matrix follows from the above discussion, Corollary 4, and Lemma 8.

Theorem 9 *Let $\widetilde{\boldsymbol{A}}$ be a noisy separable matrix of the form $\boldsymbol{A}+\boldsymbol{N}$. Assume that Assumption 1 holds for the separable matrix $\boldsymbol{A}$ in $\widetilde{\boldsymbol{A}}$. Set $\epsilon = \frac{1}{4}\sigma(1-\mu)$ where $\sigma = \sigma_r(\boldsymbol{F})$ and $\mu = \mu(\boldsymbol{K})$ for the basis and weight matrices $\boldsymbol{F}$ and $\boldsymbol{K}$ of $\boldsymbol{A}$. If $||\boldsymbol{N}||_2 < \epsilon$, Algorithm 1 for $(\widetilde{\boldsymbol{A}}, \mathrm{rank}(\boldsymbol{A}))$ returns an index set $\mathcal{I}$ such that $||\widetilde{\boldsymbol{A}}(\mathcal{I}) - \boldsymbol{F}||_2 < \epsilon$.*

In Theorem 9, let $\boldsymbol{F}^* = \widetilde{\boldsymbol{A}}(\mathcal{I})$, and $\boldsymbol{W}^*$ be an optimal solution of the convex optimization problem,

$$\text{minimize } ||\widetilde{\boldsymbol{A}}(\mathcal{I})\boldsymbol{X} - \widetilde{\boldsymbol{A}}||_F^2 \text{ subject to } \boldsymbol{X} \geq \boldsymbol{0},$$

where the matrix $\boldsymbol{X}$ of size r-by-m is the decision variable. Then, $(\boldsymbol{F}^*, \boldsymbol{W}^*)$ serves as the NMF factor of $\widetilde{\boldsymbol{A}}$. It is possible to evaluate the residual error of this factorization in a similar way to the proof of Theorem 4 by Gillis and Vavasis (2014).

Corollary 10 *Let $\boldsymbol{w}_i^*$ and $\widetilde{\boldsymbol{a}}_i$ be the column vectors of $\boldsymbol{W}^*$ and $\widetilde{\boldsymbol{A}}$, respectively. Then, $||\boldsymbol{F}^*\boldsymbol{w}_i^* - \widetilde{\boldsymbol{a}}_i||_2 < 2\epsilon$ for $i = 1, \ldots, m$.*

Proof From Assumption 1-a, the column vectors $\boldsymbol{w}_i$ of $\boldsymbol{W}$ satisfy $||\boldsymbol{w}_i||_2 \leq 1$ for $i = 1, \ldots, m$. Therefore, for $i = 1, \ldots, m$,

$$\begin{aligned} ||\boldsymbol{F}^*\boldsymbol{w}_i^* - \widetilde{\boldsymbol{a}}_i||_2 &\leq ||\boldsymbol{F}^*\boldsymbol{w}_i - \widetilde{\boldsymbol{a}}_i||_2 \\ &= ||\boldsymbol{F}^*\boldsymbol{w}_i - \boldsymbol{F}\boldsymbol{w}_i + \boldsymbol{F}\boldsymbol{w}_i - \boldsymbol{a}_i - \boldsymbol{n}_i||_2 \\ &= ||(\boldsymbol{F}^* - \boldsymbol{F})\boldsymbol{w}_i - \boldsymbol{n}_i||_2 \\ &\leq ||\boldsymbol{F}^* - \boldsymbol{F}||_2||\boldsymbol{w}_i||_2 + ||\boldsymbol{n}_i||_2 < 2\epsilon, \end{aligned}$$

where $\boldsymbol{a}_i$ and $\boldsymbol{n}_i$ denote the ith column vector of $\boldsymbol{A}$ and $\boldsymbol{N}$, respectively. ■

5. Implementation in Practice

Theorem 9 guarantees that Algorithm 1 correctly identifies the near-basis matrix of a noisy separable matrix if the noise is smaller than some level. But in the NMFs of matrices arising from practical applications, it seems that the noise level would likely exceed the level for

which the theorem is valid. In such a situation, the algorithm might generate more active points than hoped. Therefore, we need to add a selection step in which r points are selected from the active points. Also, the number of active points depends on which dimension we choose in the computation of the reduced matrix $\boldsymbol{P}$. Algorithm 1 computes the reduced matrix $\boldsymbol{P}$ of the data matrix and draws an origin-centered MVEE for the column vectors $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$ of $\boldsymbol{P}$. As we will see in Lemma 11, the number of active points depends on the dimension of $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m$. Therefore, we introduce an input parameter ρ to control the dimension. By taking account of these considerations, we design a practical implementation of Algorithm 1.

Algorithm 2 Practical Implementation of Algorithm 1

Input: $\boldsymbol{M} \in \mathbb{R}_+^{d\times m}, r \in \mathbb{N}$, and $\rho \in \mathbb{N}$.
Output: $\mathcal{I}$.

1: Run Algorithm 1 for $(\boldsymbol{M}, \rho)$. Let $\mathcal{J}$ be the index set returned by the algorithm.

2: If $|\mathcal{J}| < r$, increase ρ by 1 and go back to Step 1. Otherwise, select r elements from $\mathcal{J}$ and construct the set $\mathcal{I}$ of these elements.

One may wonder whether Algorithm 2 infinitely loops or not. In fact, we can show that under some conditions, infinite loops do not occur.

Lemma 11 *For $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m \in \mathbb{R}^\rho$, let $\mathcal{S} = \{\pm\boldsymbol{p}_1, \ldots, \pm\boldsymbol{p}_m\}$. Suppose that Assumption 2 holds. Then, $\mathbb{Q}(\mathcal{S})$ has at least ρ active points.*

Proof Consider the KKT conditions (5)-(9) for $\mathbb{Q}(\mathcal{S})$. Condition (5) requires $\Omega(\boldsymbol{z}^*)$ to be nonsingular. Since $\mathrm{rank}(\boldsymbol{P}) = \rho$ from the assumption, at least ρ nonzero z_i^* exist. Therefore, we see from condition (6) that $\mathbb{Q}(\mathcal{S})$ has at least ρ active points. ■

Proposition 12 *Suppose that we choose r such that $r \leq \mathrm{rank}(\boldsymbol{M})$. Then, Algorithm 2 terminates after a finite number of iterations.*

Proof For the active index set $\mathcal{J}$ constructed in Step 1, Lemma 11 guarantees that $|\mathcal{J}| \geq \rho$. The parameter ρ increases by 1 if $|\mathcal{J}| < r$ in Step 2 and can continue to increase up to $\rho = \mathrm{rank}(\boldsymbol{M})$. Since $r \leq \mathrm{rank}(\boldsymbol{M})$, it is necessarily to satisfy $|\mathcal{J}| \geq \rho \geq r$ after a finite number of iterations. ■

Proposition 12 implies that ρ may not be an essential input parameter since Algorithm 2 always terminates under $r \leq \mathrm{rank}(\boldsymbol{M})$ even if starting with $\rho = 1$.

There are some concerns about Algorithm 2. One is in how to select r elements from an active index set $\mathcal{J}$ in Step 2. It is possible to have various ways to make the selection. We rely on existing algorithms, such as XRAY and SPA, and perform these existing algorithms for $(\boldsymbol{M}(\mathcal{J}), \rho)$. Thus, Algorithm 1 can be regarded as a preprocessor which filters out basis vector candidates from the data points and enhance the performance of existing algorithms. Another concern is in the computational cost of solving $\mathbb{Q}$. In the next section, we describe a cutting plane strategy for efficiently performing an interior-point algorithm.

5.1 Cutting Plane Strategy for Solving $\mathbb{Q}$

Let $\mathcal{S}$ be a set of m points in $\mathbb{R}^d$. As mentioned in Section 3, $O(m^3)$ arithmetic operations are required in each iteration of an interior-point algorithm for $\mathbb{Q}(\mathcal{S})$. A cutting plane strategy is a way to reduce the number of points which we need to deal with in solving $\mathbb{Q}(\mathcal{S})$. The strategy was originally used by Sun and Freund (2004). In this section, we describe the details of our implementation.

The cutting plane strategy for solving $\mathbb{Q}$ has a geometric interpretation. It is thought of that active points contribute a lot to the drawing the MVEE for a set of points but inactive points make less of a contribution. This geometric intuition can be justified by the following proposition. Let $\boldsymbol{L}$ be a d-by-d matrix. We use the notation $\delta_{\boldsymbol{L}}(\boldsymbol{p})$ to denote $\langle \boldsymbol{p}\boldsymbol{p}^\top, \boldsymbol{L}\rangle$ for an element $\boldsymbol{p} \in \mathbb{R}^d$ of $\mathcal{S}$.

Proposition 13 *Let $\bar{\mathcal{S}}$ be a subset of $\mathcal{S}$. If an optimal solution $\bar{\boldsymbol{L}}^*$ of $\mathbb{Q}(\bar{\mathcal{S}})$ satisfies $\delta_{\bar{\boldsymbol{L}}^*}(\boldsymbol{p}) \leq 1$ for all $p \in \mathcal{S} \setminus \bar{\mathcal{S}}$, then $\bar{\boldsymbol{L}}^*$ is an optimal solution of $\mathbb{Q}(\mathcal{S})$.*

The proof is omitted since it is obvious. The proposition implies that $\mathbb{Q}(\mathcal{S})$ can be solved by using its subset $\bar{\mathcal{S}}$ instead of $\mathcal{S}$. The cutting plane strategy offers a way of finding such a $\bar{\mathcal{S}}$, in which a smaller problem $\mathbb{Q}(\bar{\mathcal{S}})$ has the same optimal solution as $\mathbb{Q}(\mathcal{S})$. In this strategy, we first choose some points from $\mathcal{S}$ and construct a set $\mathcal{S}^1$ containing these points. Let $\mathcal{S}^k$ be the set constructed in the kth iteration. In the $(k+1)$th iteration, we choose some points from $\mathcal{S} \setminus \mathcal{S}^k$ and expand $\mathcal{S}^k$ to $\mathcal{S}^{k+1}$ by adding these points to $\mathcal{S}^k$. Besides expanding, we also shrink $\mathcal{S}^k$ by discarding some points which can be regarded as useless for drawing the origin-centered MVEE. These expanding and shrinking phases play an important role in constructing a small set. Algorithm 3 describes a cutting plane strategy for solving $\mathbb{Q}(\mathcal{S})$.

Algorithm 3 Cutting Plane Strategy for Solving $\mathbb{Q}(\mathcal{S})$

Input: $\mathcal{S} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_m\}$.
Output: $\boldsymbol{L}^*$.

1: Choose an initial set $\mathcal{S}^1$ from $\mathcal{S}$ and let $k = 1$.

2: Solve $\mathbb{Q}(\mathcal{S}^k)$ and find the optimal solution $\boldsymbol{L}^k$. If $\delta_{\boldsymbol{L}^k}(\boldsymbol{p}) \leq 1$ holds for all $\boldsymbol{p} \in \mathcal{S} \setminus \mathcal{S}^k$, let $\boldsymbol{L}^* = \boldsymbol{L}^k$, and stop.

3: Choose a subset $\mathcal{F}$ of $\mathcal{S}^k$ and a subset $\mathcal{G}$ of $\{\boldsymbol{p} \in \mathcal{S} \setminus \mathcal{S}^k : \delta_{\boldsymbol{L}^k}(\boldsymbol{p}) > 1\}$. Update $\mathcal{S}^k$ as $\mathcal{S}^{k+1} = (\mathcal{S}^k \setminus \mathcal{F}) \cup \mathcal{G}$ and increase k by 1. Then, go back to Step 2.

Now, we give a description of our implementation of Algorithm 3. To construct the initial set $\mathcal{S}^1$ in Step 1, our implementation employs the algorithm used in the papers (Kumar and Yildirim, 2005; Todd and Yildirim, 2007; Ahipasaoglu et al., 2008). The algorithm constructs a set $\mathcal{S}^1$ by greedily choosing $2d$ points in a step-by-step manner such that the convex hull is a d-dimensional crosspolytope containing as many points in $\mathcal{S}$ as possible. We refer the reader to Algorithm 3.1 of Kumar and Yildirim (2005) for the precise description.

To shrink and expand $\mathcal{S}^k$ in Step 3, we use a shrinking threshold parameter θ such that $\theta < 1$, and an expanding size parameter η such that $\eta \geq 1$. These parameters are set before running Algorithm 3. For shrinking, we construct $\mathcal{F} = \{\boldsymbol{p} \in \mathcal{S}^k : \delta_{\boldsymbol{L}^k}(\boldsymbol{p}) \leq \theta\}$ by using θ.

For expanding, we arrange the points of $\{\boldsymbol{p} \in \mathcal{S} \setminus \mathcal{S}^k : \delta_{\boldsymbol{L}^k}(\boldsymbol{p}) > 1\}$ in descending order, as measured by $\delta_{\boldsymbol{L}^k}(\cdot)$, and construct $\mathcal{G}$ by choosing the top $(m-2d)/\eta$ points. If the set $\{\boldsymbol{p} \in \mathcal{S} \setminus \mathcal{S}^k : \delta_{\boldsymbol{L}^k}(\boldsymbol{p}) > 1\}$ has less than $(m-2d)/\eta$ points, we choose all the points and construct $\mathcal{G}$.

6. Experiments

We experimentally compared Algorithm 2 with SPA and the variants of XRAY. These two existing algorithms were chosen because their studies (Bittorf et al., 2012; Gillis and Luce, 2013; Kumar et al., 2013) report that they outperform AGKM and Hottopixx, and scale to the problem size. Two types of experiments were conducted: one is the evaluation for the robustness of the algorithms to noise on synthetic data sets, and the other is the application of the algorithms to clustering of real-world document corpora.

We implemented Algorithm 2, and three variants of XRAY, "max", "dist" and "greedy", in MATLAB. We put Algorithm 3 in Algorithm 2 so it would solve $\mathbb{Q}$ efficiently. The software package SDPT3 (Toh et al., 1999) was used for solving $\mathbb{Q}(\mathcal{S}^k)$ in Step 2 of Algorithm 3. The shrinking parameter θ and expanding size parameter η were set as 0.9999 and 5, respectively. The implementation of XRAY formulated the computation of the residual matrix $\boldsymbol{R} = \boldsymbol{A}(\mathcal{I}_k)\boldsymbol{X}^* - \boldsymbol{A}$ as a convex optimization problem,

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X} \geq \boldsymbol{0}} ||\boldsymbol{A}(\mathcal{I}_k)\boldsymbol{X} - \boldsymbol{A}||_F^2.$$

For the implementation of SPA (Gillis and Vavasis, 2014), we used code from the first author's website. Note that SPA and XRAY are sensitive to the normalization of the column vectors of the data matrix (see, for instance, Kumar et al., 2013), and for this reason, we used a data matrix whose column vectors were not normalized. All experiments were done in MATLAB on a 3.2 GHz CPU processor and 12 GB memory.

We will use the following abbreviations to represent the variants of algorithms. For instance, Algorithm 2 with SPA for an index selection of Step 2 is referred to as ER-SPA. Also, the variant of XRAY with "max" selection policy is referred to as XRAY(max).

6.1 Synthetic Data

Experiments were conducted for the purpose of seeing how well Algorithm 2 could improve the robustness of SPA and XRAY to noise. Specifically, we compared it with SPA, XRAY(max), XRAY(dist), and XRAY(greedy). The robustness of algorithm was measured by a recovery rate. Let $\mathcal{I}$ be an index set of basis vectors in a noisy separable matrix, and $\mathcal{I}^*$ be an index set returned by an algorithm. The recovery rate is the ratio given by $|\mathcal{I} \cap \mathcal{I}^*| \ / \ |\mathcal{I}|$.

We used synthetic data sets of the form $\boldsymbol{F}(\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi} + \boldsymbol{N}$ with $d = 250$, $m = 5,000$, and $r = 10$. The matrices $\boldsymbol{F}, \boldsymbol{K}, \boldsymbol{\Pi}$ and $\boldsymbol{N}$ were synthetically generated as follows. The entries of $\boldsymbol{W} \in \mathbb{R}_+^{d \times r}$ were drawn from a uniform distribution on the interval $[0, 1]$. The column vectors of $\boldsymbol{K} \in \mathbb{R}_+^{r \times \ell}$ were from a Dirichlet distribution whose r parameters were uniformly from the interval $[0, 1]$. The permutation matrix $\boldsymbol{\Pi}$ was randomly generated. The entries of the noise matrix $\boldsymbol{N} \in \mathbb{R}^{d \times m}$ were from a normal distribution with mean 0 and standard deviation δ. The parameter δ determined the intensity of the noise, and it was chosen from

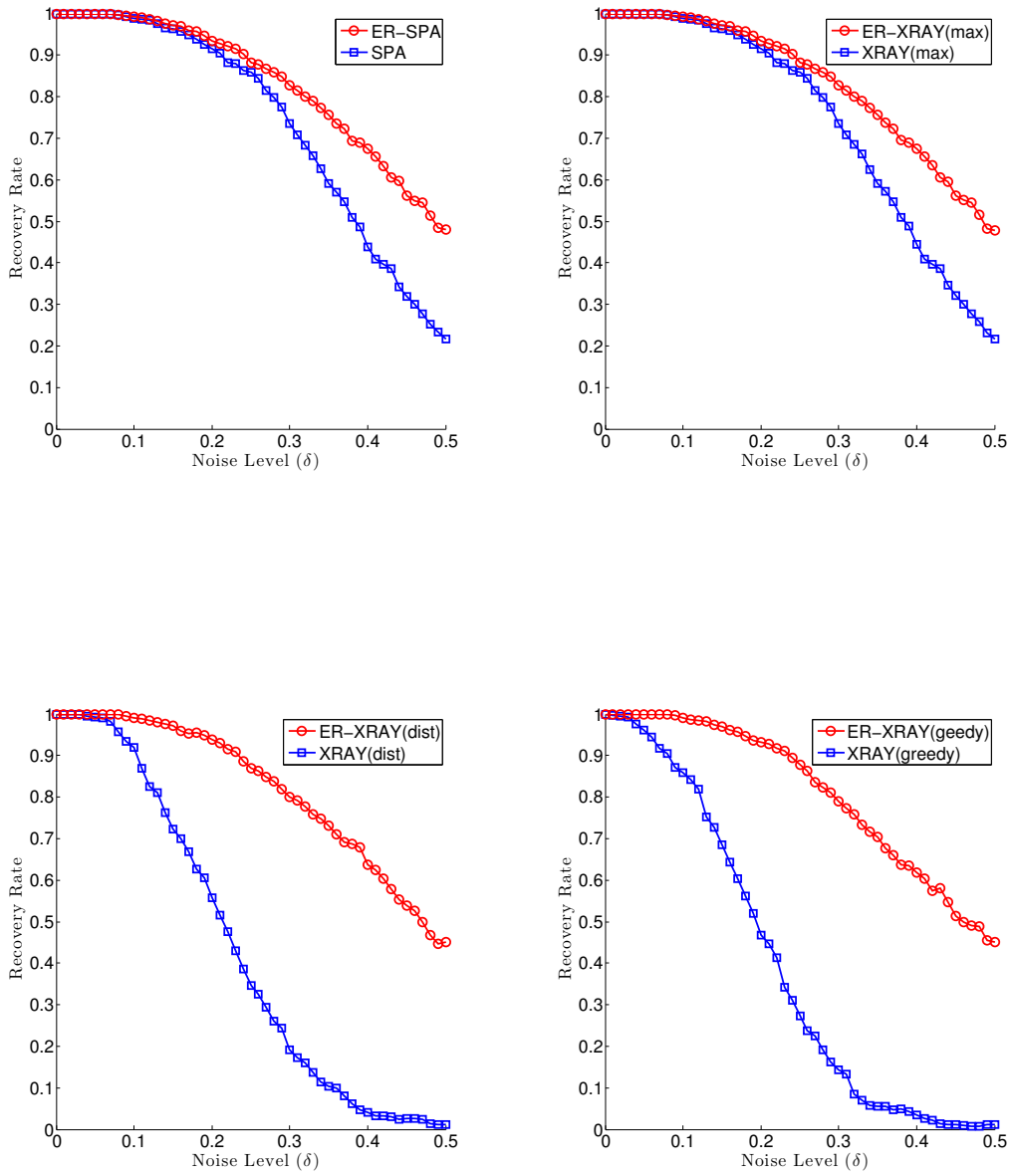

Figure 3: Comparison of the recovery rates of Algorithm 2 with SPA and XRAY.

Recovery rate	100%	90%	80%	70%
ER-SPA	0.06	0.24	0.32	0.37
SPA	0.05	0.21	0.27	0.31
ER-XRAY(max)	0.06	0.24	0.32	0.37
XRAY(max)	0.05	0.21	0.27	0.31
ER-XRAY(dist)	0.07	0.23	0.29	0.36
XRAY(dist)	0.03	0.10	0.13	0.16
ER-XRAY(greedy)	0.07	0.23	0.29	0.35
XRAY(greedy)	0.00	0.08	0.12	0.14

Table 1: Maximum values of noise level δ for different recovery rates in percentage.

0 to 0.5 in 0.01 increments. A single data set consisted of 51 matrices with various amounts of noise, and we made 50 different data sets. Algorithm 2 was performed in the setting that $\boldsymbol{M}$ is a matrix in the data set and r and ρ are each 10.

Figure 3 depicts the average recovery rate on the 50 data sets for Algorithm 2, SPA and XRAY. Table 1 summarizes the maximum values of noise level δ for different recovery rates in percentage. The noise level was measured by 0.01, and hence, for instance, the entry "0.00" at XRAY(greedy) for 100% recovery rate means that the maximum value is in the interval $[0.00, 0.01)$. We see from the figure that Algorithm 2 improved the recovery rates of the existing algorithms. In particular, the recovery rates of XRAY(dist) and XRAY(greedy) rapidly decrease as the noise level increases, but Algorithm 2 significantly improved them. Also, the figure shows that Algorithm 2 tended to slow the decrease in the recovery rate. We see from the table that Algorithm 2 is more robust to noise than SPA and XRAY.

Table 2 summarizes the average number of active points and elapsed time for 50 data sets taken by Algorithm 2 with $\delta = 0, 0.25$ and 0.5. We read from the table that the elapsed time increases with the number of active points. The average elapsed times of SPA, XRAY(max), XRAY(dist), and XRAY(greedy) was respectively 0.03, 1.18, 16.80 and 15.85 in seconds. Therefore, we see that the elapsed time of Algorithm 2 was within a reasonable range.

δ	Active points	Elapsed time (second)			
		ER-SPA	ER-XRAY(max)	ER-XRAY(dist)	ER-XRAY(greedy)
0	10	1.05	1.07	1.07	1.07
0.25	12	3.08	3.10	3.10	3.10
0.5	23	4.70	4.71	4.71	4.71

Table 2: Average number of active points and elapsed time of Algorithm 2.

6.2 Application to Document Clustering

Consider a set of d documents. Let m be the total number of words appearing in the document set. We represent the documents by a bag-of-words. That is, the ith document is represented as an m-dimensional vector $\boldsymbol{a}_i$, whose elements are the appearance frequencies of words in the document. A document vector $\boldsymbol{a}_i$ can be assumed to be generated by a

convex combination of several topic vectors $\boldsymbol{w}_1, \dots \boldsymbol{w}_r$. This type of generative model has been used in many papers (for instance, Xu et al., 2003; Shahnaz et al., 2006; Arora et al., 2012b, 2013; Ding et al., 2013; Kumar et al., 2013).

Let $\boldsymbol{W}$ be an r-by-m topic matrix such that $\boldsymbol{w}_1^\top, \dots, \boldsymbol{w}_r^\top$ are stacked from top to bottom and are of the form $(\boldsymbol{w}_1; \dots; \boldsymbol{w}_r)$. The model allows us to write a document vector in the form $\boldsymbol{a}_i^\top = \boldsymbol{f}_i^\top \boldsymbol{W}$ by using a coefficient vector $\boldsymbol{f}_i \in \mathbb{R}^r$ such that $\boldsymbol{e}^\top \boldsymbol{f}_i = 1$ and $\boldsymbol{f}_i \geq \boldsymbol{0}$. This means that we have $\boldsymbol{A} = \boldsymbol{F}\boldsymbol{W}$ for a document-by-word matrix $\boldsymbol{A} = (\boldsymbol{a}_1; \dots; \boldsymbol{a}_d) \in \mathbb{R}_+^{d \times m}$, a coefficient matrix $\boldsymbol{F} = (\boldsymbol{f}_1; \dots; \boldsymbol{f}_d) \in \mathbb{R}_+^{d \times r}$, and a topic matrix $\boldsymbol{W} = (\boldsymbol{w}_1; \dots; \boldsymbol{w}_r) \in \mathbb{R}_+^{r \times m}$. In the same way as the papers (Arora et al., 2012b, 2013; Ding et al., 2013; Kumar et al., 2013), we assume that a document-by-word matrix $\boldsymbol{A}$ is separable. This requires that $\boldsymbol{W}$ is of $(\boldsymbol{I}, \boldsymbol{K})\boldsymbol{\Pi}$, and it means that each topic has an *anchor word*. An anchor word is a word that is contained in one topic but not contained in the other topics. If an anchor word is found, it suggests that the associated topic exists.

Algorithms for Problem 1 can be used for clustering documents and finding topics for the above generative model. The algorithms for a document-word matrix $\boldsymbol{A}$ return an index set $\mathcal{I}$. Let $\boldsymbol{F} = \boldsymbol{A}(\mathcal{I})$. The row vector elements $f_{i1}, \dots, f_{ir}$ of $\boldsymbol{F}$ can be thought of as the contribution rate of topics $\boldsymbol{w}_1, \dots, \boldsymbol{w}_r$ for generating a document $\boldsymbol{a}_i$. The highest value f_{ij^*} among the elements implies that the topic $\boldsymbol{w}_{j^*}$ contributes the most to the generation of document $\boldsymbol{a}_i$. Hence, we assign document $\boldsymbol{a}_i$ to a cluster having the topic $\boldsymbol{w}_{j^*}$. There is an alternative to using $\boldsymbol{F}$ for measuring the contribution rates of the topics. Step 1 of Algorithm 1 produces a rank-r approximation matrix $\boldsymbol{A}^r$ to $\boldsymbol{A}$ as a by-product. Let $\boldsymbol{F}' = \boldsymbol{A}^r(\mathcal{I})$, and use it as an alternative to $\boldsymbol{F}$. We say that clustering with $\boldsymbol{F}$ is clustering with the original data matrix, and that clustering with $\boldsymbol{F}'$ is clustering with a low-rank approximation data matrix.

Experiments were conducted in the purpose of investigating clustering performance of algorithms and also checking whether meaningful topics could be extracted. To investigate the clustering performance, we used only SPA since our experimental results implied that XRAY would underperform. We assigned the values of the document-word matrix on the basis of the tf-idf weighting scheme, for which we refer the reader to Manning et al. (2008), and normalized the row vectors to the unit 1-norm.

To evaluate the clustering performance, we measured the accuracy (AC) and normalized mutual information (NMI). These measures are often used for this purpose (see, for instance, Xu et al., 2003; Manning et al., 2008). Let $\Omega_1, \dots, \Omega_r$ be the manually classified classes and $\mathcal{C}_1, \dots, \mathcal{C}_r$ be the clusters constructed by an algorithm. Both Ω_i and $\mathcal{C}_j$ are the subsets of the document set $\{\boldsymbol{a}_1, \dots, \boldsymbol{a}_m\}$ such that each subset does not share any documents and the union of all subsets coincides with the document set. AC is computed as follows. First, compute the correspondence between classes $\Omega_1, \dots, \Omega_r$ and clusters $\mathcal{C}_1, \dots, \mathcal{C}_r$ such that the total number of common documents $\Omega_i \cap \mathcal{C}_j$ is maximized. This computation can be done by solving an assignment problem. After that, rearrange the classes and clusters in the obtained order and compute

$$\frac{1}{d} \sum_{k=1}^{r} |\Omega_k \cap \mathcal{C}_k|.$$

This value is the AC for the clusters constructed by an algorithm. NMI is computed as

$$\frac{I(\Omega, \mathcal{C})}{\frac{1}{2}(E(\Omega) + E(\mathcal{C}))}.$$

I and E denote the mutual information and entropy for the class family Ω and cluster family $\mathcal{C}$ where $\Omega = \{\Omega_1, \ldots, \Omega_r\}$ and $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_r\}$. We refer the reader to Section 16.3 of Manning et al. (2008) for the precise forms of I and E.

Two document corpora were used for the clustering-performance evaluation: Reuters-21578 and 20 Newsgroups. These corpora are publicly available from the UCI Knowledge Discovery in Databases Archive (`http://kdd.ics.uci.edu`). In particular, we used the data preprocessing of Deng Cai, in which multiple classes are discarded. The data sets are available from the website (`http://www.cad.zju.edu.cn/home/dengcai`). The Reuters-21578 corpus consists of 21,578 documents appearing in the Reuters newswire in 1987, and these documents are manually classified into 135 classes. The text corpus is reduced by the preprocessing to 8,293 documents in 65 classes. Furthermore, we cut off classes with less than 5 documents. The resulting corpus contains 8,258 documents with 18,931 words in 48 classes, and the sizes of the classes range from 5 to 3,713. The 20 Newsgroups corpus consists of 18,846 documents with 26,213 words appearing in 20 different newsgroups. The size of each class is about 1,000.

We randomly picked some classes from the corpora and evaluated the clustering performance 50 times. Algorithm 2 was performed in the setting that $\boldsymbol{M}$ is a document-word matrix and r and ρ each are the number of classes. In clustering with a low-rank approximation data matrix, we used the rank-r approximation matrix to a document-word matrix.

	AC				NMI			
	Original		Low-rank approx.		Original		Low-rank approx.	
# Classes	ER-SPA	SPA	ER-SPA	SPA	ER-SPA	SPA	ER-SPA	SPA
6	0.605	0.586	0.658	0.636	0.407	0.397	0.532	0.466
8	0.534	0.539	0.583	0.581	0.388	0.387	0.491	0.456
10	0.515	0.508	0.572	0.560	0.406	0.393	0.511	0.475
12	0.482	0.467	0.532	0.522	0.399	0.388	0.492	0.469

Table 3: (Reuters-21578) Average AC and NMI of ER-SPA and SPA with the original data matrix and low-rank approximation data matrix.

Tables 3 and 4 show the results for Reuters-21578 and 20 Newsgroups, respectively. They summarize the average ACs and NMIs of ER-SPA and SPA. The column with "# Classes" lists the number of classes we chose. The columns labeled "Original" and "Low-rank approx." are respectively the averages of the corresponding clustering measurements with the original data matrix and low-rank approximation data matrix. The tables suggest that clustering with a low-rank approximation data matrix performed better than clustering with the original data matrix. We see from Table 3 that ER-SPA could achieve improvements in the AC and NMI of SPA on Reuters-21578 when the clustering was done with a

	AC				NMI			
	Original		Low-rank approx.		Original		Low-rank approx.	
# Classes	ER-SPA	SPA	ER-SPA	SPA	ER-SPA	SPA	ER-SPA	SPA
6	0.441	0.350	0.652	0.508	0.314	0.237	0.573	0.411
8	0.391	0.313	0.612	0.474	0.306	0.242	0.555	0.415
10	0.356	0.278	0.559	0.439	0.291	0.228	0.515	0.397
12	0.319	0.240	0.517	0.395	0.268	0.205	0.486	0.372

Table 4: (20 Newsgroups) Average AC and NMI of ER-SPA and SPA with the original data matrix and low-rank approximation data matrix.

low-rank approximation data matrix. Table 4 indicates that ER-SPA outperformed SPA in AC and NMI on 20 Newsgroups.

Finally, we compared the topics obtained by ER-SPA and SPA. We used the BBC corpus of Greene and Cunningham (2006), which is available from the website (`http://mlg.ucd.ie/datasets/bbc.html`). The documents in the corpus have been subjected by preprocessed such as stemming, stop-word removal, and low word frequency filtering. It consists of 2,225 documents with 9,636 words that appeared on the BBC news website in 2004-2005. The documents were news on 5 topics: "business", "entertainment", "politics", "sport" and "tech".

AC		NMI	
ER-SPA	SPA	ER-SPA	SPA
0.939	0.675	0.831	0.472

Table 5: AC and NMI of ER-SPA and SPA with low-rank approximation data matrix for BBC.

	Anchor word	1	2	3	4	5
ER-SPA	film	award	best	oscar	nomin	actor
SPA	film	award	best	oscar	nomin	star
ER-SPA	mobil	phone	user	softwar	microsoft	technolog
SPA	mobil	phone	user	microsoft	music	download
ER-SPA	bank	growth	economi	price	rate	oil
SPA	bank	growth	economi	price	rate	oil
ER-SPA	game	plai	player	win	england	club
SPA	fiat	sale	profit	euro	japan	firm
ER-SPA	elect	labour	parti	blair	tori	tax
SPA	blog	servic	peopl	site	firm	game

Table 6: Anchor words and top-5 frequent words in topics grouped by ER-SPA and SPA for BBC.

Table 5 shows the ACs and NMIs of ER-SPA and SPA on the low-rank approximation data matrix for the BBC corpus. The table indicates that the AC and NMI of ER-SPA are higher than those of SPA. Table 6 summarizes the words in the topics obtained by ER-SPA and SPA. The topics were computed by using a low-rank approximation data matrix. The table lists the anchor word and the 5 most frequent words in each topic from left to right. We computed the correspondence between topics obtained by ER-SPA and SPA and grouped the topics for each algorithm. Concretely, we measured the 2-norm of each topic vector and computed the correspondence by solving an assignment problem. We can see from the table that the topics obtained by these two algorithms are almost the same from the first to the third panel, and they seem to correspond to "entertainment", "tech" and "business". The topics in the fourth and fifth panels, however, are different. The topic in the fifth panel by ER-SPA seems to correspond to "politics". In contrast, it is difficult to find the topic corresponding to "politics" in the panels by SPA. These show that ER-SPA could extract more recognizable topics than SPA.

Remark 14 *Sparsity plays an important role in computing the SVD for a large document corpus. In general, a document-word matrix arising from a text corpus is quite sparse. Our implementation of Algorithm 2 used the MATLAB command* `svds` *that exploits the sparsity of a matrix in the SVD computation. The implementation could work on all data of 20 Newsgroups corpus, which formed a document-word matrix of size 18,846-by-26,213.*

7. Concluding Remarks

We presented Algorithm 1 for Problem 1 and formally showed that it has correctness and robustness properties. Numerical experiments on synthetic data sets demonstrated that Algorithm 2, which is the practical implementation of Algorithm 1, is robustness to noise. The robustness of the algorithm was measured in terms of the recovery rate. The results indicated that Algorithm 2 can improve the recovery rates of SPA and XRAY. The algorithm was then applied to document clustering. The experimental results implied that it outperformed SPA and extracted more recognizable topics.

We will conclude by suggesting a direction for future research. Algorithm 2 needs to do two computations: one is the SVD of the data matrix and the other is the MVEE for a set of reduced-dimensional data points. It would be ideal to have a single computation that could be parallelized. The MVEE computation requires that the convex hull of data points is full-dimensional. Hence, the SVD computation should be carried out on data points. However, if we could devise an alternative convex set for MVEE, it would possible to avoid SVD computation. It would be interesting to investigate the possibility of algorithms that find near-basis vectors by using the other type of convex set for data points.

Acknowledgments

The author would like to thank Akiko Takeda of the University of Tokyo for her insightful and enthusiastic discussions, and thank the referees for careful reading and helpful suggestions that considerably improved the quality of this paper.

References

S. D. Ahipasaoglu, P. Sun, and M. J. Todd. Linear convergence of a modified Frank-Wolfe algorithm for computing minimum-volume enclosing ellipsoids. *Optimization Methods and Software*, 23(1):5–19, 2008.

S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization – Provably. In *Proceedings of the 44th Symposium on Theory of Computing (STOC)*, pages 145–162, 2012a.

S. Arora, R. Ge, and A. Moitra. Learning topic models – Going beyond SVD. In *Proceedings of the 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1–10, 2012b.

S. Arora, R. Ge, Y. Halpern, D. Mimno, and A. Moitra. A practical algorithm for topic modeling with provable guarantees. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

V. Bittorf, B. Recht, C. Re, and J. A. Tropp. Factoring nonnegative matrices with linear programs. In *Advances in Neural Information Processing Systems 25 (NIPS)*, pages 1223–1231, 2012.

A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation*. Wiley, 2009.

W. Ding, M. H. Rohban, P. Ishwar, and V. Saligrama. Topic discovery through data dependent and random projections. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems 16 (NIPS)*, pages 1141–1148, 2003.

N. Gillis. Robustness analysis of Hottopixx, a linear programming model for factoring nonnegative matrices. *SIAM Journal on Matrix Analysis and Applications*, 34(3):1189–1212, 2013.

N. Gillis and R. Luce. Robust near-separable nonnegative matrix factorization using linear optimization. arXiv:1302.4385v1, 2013.

N. Gillis and S. A. Vavasis. Fast and robust recursive algorithms for separable nonnegative matrix factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):698–714, 2014.

G. H. Golub and C. F. Van Loan. *Matrix Computation*. The Johns Hopkins University Press, 3rd edition, 1996.

P. Gong and C. Zhang. Efficient nonnegative matrix factorization via projected newton method. *Pattern Recognition*, 45(9):3557–3565, 2012.

D. Greene and P. Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23th International Conference on Machine Learning (ICML)*, 2006.

L. G. Khachiyan. Rounding of polytopes in the real number model of computation. *Mathematics of Operations Research*, 21(2):307–320, 1996.

H. Kim and H. Park. Non-negative matrix factorization based on alternating non-negativity constrained least squares and active set method. *SIAM Journal on Matrix Analysis and Applications*, 30(2):713–730, 2008.

H. Kim and H. Park. Fast nonnegative matrix factorization: An active-set-like method and comparisons. *SIAM Journal on Scientific Computing*, 33(6):3261–3281, 2011.

J. Kim, Y. He, and H. Park. Algorithms for nonnegative matrix and tensor factorizations: a unified view based on block coordinate descent framework. *Journal of Global Optimization*, 58(2):285–319, 2014.

A. Kumar, V. Sindhwani, and P. Kambadur. Fast conical hull algorithms for near-separable non-negative matrix factorization. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

P. Kumar and E. A. Yildirim. Minimum-volume enclosing ellipsoids and core sets. *Journal of Optimization Theory and Applications*, 126(1), 2005.

D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13 (NIPS)*, pages 556–562, 2001.

C.-J. Lin. Projected gradient methods for non-negative matrix factorization. *Neural Computation*, 19(10):2756–2779, 2007.

C. D. Manning, P. Raghavan, and H. Schuetze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

L. Miao and H. Qi. Endmember extraction from highly mixed data using minimum volume constrained nonnegative matrix factorization. *IEEE Transactions on Geoscience and Remote Sensing*, 45(2):765–777, 2007.

J. M. P. Nascimento and J. M. B. Dias. Vertex component analysis: A fast algorithm to unmix hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43 (4):898–910, 2005.

F. Shahnaz, M. W. Berry, V. P. Pauca, and R. J. Plemmons. Document clustering using nonnegative matrix factorization. *Information Processing and Management*, 42(2):373–386, 2006.

P. Sun and R. M. Freund. Computation of minimum-volume covering ellipsoids. *Operations Research*, 52(5):690–706, 2004.

M. J. Todd and E. A. Yildirim. On Khachiyan's algorithm for the computation of minimum-volume enclosing ellipsoids. *Discrete Applied Mathematics*, 155(13):1731–1744, 2007.

K.-C. Toh. Primal-dual path-following algorithms for determinant maximization problems with linear matrix inequalities. *Computational Optimization and Applications*, 14(3): 309–330, 1999.

K.-C. Toh, M. J. Todd, and R. H. Tütüncü. SDPT3 – a MATLAB software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

T. Tsuchiya and Y. Xia. An extension of the standard polynomial-time primal-dual path-following algorithm to the weighted determinant maximization problem with semidefinite constraints. *Pacific Journal of Optimization*, 3(1):165–182, 2007.

L. Vandenberghe, S. Boyd, and S. P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499–533, 1998.

S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal of Optimization*, 20(3):1364–1377, 2009.

W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 267–273, 2003.