CLUSTERING AND LATENT SEMANTIC INDEXING ASPECTS OF THE NONNEGATIVE MATRIX FACTORIZATION

ANDRI MIRZAL*

Abstract. This paper provides a theoretical support for clustering aspect of the nonnegative matrix factorization (NMF). By utilizing the Karush-Kuhn-Tucker optimality conditions, we show that NMF objective is equivalent to graph clustering objective, so clustering aspect of the NMF has a solid justification. Different from previous approaches which usually discard the nonnegativity constraints, our approach guarantees the stationary point being used in deriving the equivalence is located on the feasible region in the nonnegative orthant. Additionally, since clustering capability of a matrix decomposition technique can sometimes imply its latent semantic indexing (LSI) aspect, we will also evaluate LSI aspect of the NMF by showing its capability in solving the synonymy and polysemy problems in synthetic datasets. And more extensive evaluation will be conducted by comparing LSI performances of the NMF and the singular value decomposition (SVD)—the standard LSI method—using some standard datasets.

Key words. bound-constrained optimization, clustering method, nonnegative matrix factorization, Karush-Kuhn-Tucker optimality conditions, latent semantic indexing.

AMS subject classifications. 15A23, 68R10.

1. Introduction. Nonnegative datasets are everywhere; from term by document matrix induced from a document corpus [1, 2], gene expression datasets [3], pixels in digital images [4], disease patterns [5], to spectral signatures from astronomical spectrometers [6] among others. Even though diverse, they have one thing in common: all can be represented by using nonnegative matrices induced from the datasets. This allows many well-established mathematical techniques to be applied in order to analyze the datasets.

There are many common tasks associated with these datasets, for example: grouping the similar data points (clustering), finding patterns in the datasets, identifying important or interesting features, and finding sets of relevant data points to queries (information retrieval). In this paper, we will focus on two tasks: clustering and latent semantic indexing—a technique that can be used for improving recall and precision of an information retrieval (IR) system.

1.1. Clustering. Clustering is the task of assigning data points into clusters such that similar points are in the same clusters and dissimilar points are in the...
different clusters. There are many types of clustering, for example supervised/unsupervised, hierarchical/partitional, hard/soft, and one-way/many-way (two-way clustering is known as co-clustering or bi-clustering) among others. In this paper, clustering term refers to unsupervised, partitional, hard, and one-way clustering. Further, the number of cluster is given beforehand.

The NMF as a clustering method can be traced back to the work by Lee & Seung [4]. But, the first work that explicitly demonstrates it is the work by Xu et al. [1] in which they show that the NMF outperforms the spectral methods in term of purity and mutual information measures for Reuters and TDT2 datasets.

Clustering aspect of the NMF, even though numerically well studied, is not theoretically well explained. Usually this aspect is explained by showing the equivalence between NMF objective to either k-means clustering objective [7, 8] or spectral clustering objective [2]. The problem with the first approach is there is no obvious way to incorporate the nonnegativity constraints into k-means clustering objective. And the problem with the second approach is it discards the nonnegativity constraints, thus is equivalent to finding stationary points on unbounded region. Accordingly, the NMF which is a bound-constrained optimization turns into an unbounded optimization, so there is no guarantee the stationary point being utilized in proving the equivalence is located on the feasible region indicated by the constraints.

In the first part of this paper, we will provide a theoretical support for clustering aspect of the NMF by analyzing the objective at the stationary point using the Karush-Kuhn-Tucker (KKT) conditions without setting the KKT multipliers to zeros. Thus, the stationary point under investigation is guaranteed to be located on the feasible region.

1.2. Latent semantic indexing. Latent semantic indexing (LSI) is a method introduced by Deerwester et al. [9] to improve recall and precision of an IR system using truncated singular value decomposition (SVD) of the term-by-document matrix to reveal hidden relationship between documents by indexing terms that are present in the similar documents and weakening the influences of terms that are mutually present in the dissimilar documents. The first capability can solve the synonymy—different words with similar meaning—problem, and the second capability can solve the polysemy—words with multiple unrelated meanings—problem. Thus, LSI not only is able to retrieve relevant documents that do not contain terms in the query, but also can filter out irrelevant documents that contain terms in the query.

LSI aspect of the NMF is not well studied. There are some works that discuss the relationship between the NMF and probabilistic LSI, e.g., [10, 11]. But the emphasize is in clustering capability of probabilistic LSI, not LSI aspect of the NMF. Motivated by the SVD which is the standard method in clustering and LSI, in the second part of
2. The nonnegative matrix factorization. The NMF was popularized by the work of Lee & Seung [4] in which they showed that this technique can be used to learn parts of faces and semantic features of text. Previously, it has been studied under the term positive matrix factorization [12, 13]. Mathematically, the NMF is a technique that decomposes a nonnegative data matrix into a pair of other nonnegative matrices:

$$A \approx BC,$$  \hspace{1cm} (2.1)

where $A \in \mathbb{R}^{M \times N} = [a_1, \ldots, a_N]$ denotes the data matrix, $B \in \mathbb{R}^{M \times K} = [b_1, \ldots, b_K]$ denotes the basis matrix, $C \in \mathbb{R}^{K \times N} = [c_1, \ldots, c_N]$ denotes the coefficient matrix, and $K$ denotes the number of factors which usually is chosen so that $K \ll \min(M, N)$. Note that the definitions of $A$, $B$, and $C$ are chosen to simplify the interpretation of the NMF.

To compute $B$ and $C$, usually eq. (2.1) is rewritten into a minimization problem in Frobenius norm.

$$\min_{B, C} J(B, C) = \frac{1}{2} \|A - BC\|^2_F \text{ s.t. } B \geq 0, C \geq 0. \hspace{1cm} (2.2)$$

In addition to the usual Frobenius norm, family of Bregman divergences—which Frobenius norm and Kullback-Leibler divergence are part of it—can also be used as the distance measures. Detailed discussion on Bregman divergences can be found in, e.g., ref. [14]. In this work, we will consider only Frobenius norm.

3. Limit points of the sequences generated by NMF algorithms. All NMF algorithms are formulated in the alternating fashion, fixing one matrix while solving the other (the popular Lee & Seung algorithms [15] and their derivatives, e.g., [1, 5, 16, 17] also use the alternating strategy, but cannot be represented by generic algorithm below). This strategy is employed because the NMF is nonconvex with respect to $B$ and $C$, but is convex with respect to $B$ or $C$ [18]. Thus, the alternating strategy transforms NMF problem into a pair of convex subproblems. Transforming a nonconvex problem into the corresponding convex subproblems is a common practice in optimization researches because: (1) convex optimization is more tractable, (2) usually convex methods are more efficient, (3) any local optimum is necessarily a global optimum, and (4) the algorithms are easy to initialize [19].

Algorithm 1 solves NMF problem in the alternating fashion which will generate a solution sequence $\{B^{(i)}, C^{(i)}\}_{i=0}^L$. This algorithm is known as alternating nonnegativity-constrained least square (ANLS) algorithm, and usually is solved by
Algorithm 1 Generic algorithm for the NMF based on ANLS.

Initialization: $C^0 \geq 0$.

for $l = 0, \ldots$ do

\[
B^{(l+1)} \leftarrow \arg \min_{B \geq 0} \frac{1}{2} \| A - BC^{(l)} \|_F^2, \quad (3.1)
\]

\[
C^{(l+1)} \leftarrow \arg \min_{C \geq 0} \frac{1}{2} \| A - B^{(l+1)} C \|_F^2, \quad (3.2)
\]

end for

decomposing each subproblem into the corresponding nonnegativity-constrained least square (NNLS) problems, where there are many algorithms that guarantee the global-optimality of the NNLS problems. The following equations are the NNLS versions of the ANLS in algorithm 1:

\[
\hat{b}^{T(l+1)}_m \leftarrow \arg \min_{\hat{b}^{T(l)}_m \geq 0} \frac{1}{2} \| \hat{a}^{T(l)}_m - C^{T(l)} \hat{b}^{T(l)}_m \|_F^2, \quad \forall m, \quad (3.3)
\]

\[
c^{(l+1)}_n \leftarrow \arg \min_{c^{(l)}_n \geq 0} \frac{1}{2} \| a_n - B^{(l+1)} c^{(l)}_n \|_F^2, \quad \forall n, \quad (3.4)
\]

where $\hat{x}_i$ is the $i$-th row of $X$.

According to Grippo & Sciandrone [20], any limit point of $\{B^{(l)}, C^{(l)}\}_{l=0}^L$—generated by any ANLS algorithm that optimally solves the convex subproblem eq. 3.1 and eq. 3.2—is a stationary point. And such ANLS based NMF algorithms exist, e.g., [17, 18, 21, 22, 23, 24], therefore there is guarantee that the stationary points are reachable. And as ANLS is the building block for ANLS, any NNLS algorithm that guarantees to find optimal solutions of eq. 3.3 and eq. 3.4, e.g., [25, 26, 27] can also be employed to search for the stationary points. And as will be shown in section 4 NMF objective (eq. 2.2) implicitly puts upper bounds on the feasible region (the lower bounds are explicit: the nonnegativity constraints). Thus the NMF is bound-constrained optimization problem, consequently $\{B^{(l)}, C^{(l)}\}_{l=0}^L$ has at least one limit point [18]. This completes the conditions for any NMF algorithm that optimally solves subproblem eq. 3.1 and eq. 3.2 to have convergence guarantee.

4. Clustering aspect of the NMF. This section is the first part of this paper in which a theoretical framework for supporting clustering aspect of the NMF will be provided. The strict KKT optimality conditions will be utilized to derive the equivalence between NMF objective to graph clustering objective. Unlike previous approaches where the KKT multipliers are set to zeros [7, 28, 29, 30], we will make no
assumption about the KKT multipliers, thus the stationary point under investigation is guaranteed to be located on the feasible region in the nonnegative orthant.

We will also show that the feasible region is bounded, with the lower bounds are explicitly bounded by the nonnegativity constraints, and the upper bounds are implicitly bounded by the objective. As stated in section 3 the boundedness of the feasible region is the necessary condition for guaranteeing the existence of limit point of \( \{B_l^{(i)}, C_l^{(i)}\}_{l=0}^L \). And for interpretability reason, the data matrix \( A \) will be considered as a feature-by-item data matrix unless stated differently.

The following proposition gives the theoretical support for clustering aspect of the NMF.

**Proposition 4.1.** Minimizing the following objective

\[
\min_{B,C} J(B, C) = \frac{1}{2} \|A - BC\|_F^2
\]

s.t. \( B \geq 0, C \geq 0 \),

leads to the feature clustering indicator matrix \( B \) and the item clustering indicator matrix \( C \).

**Proof.**

\[
\|A - BC\|_F^2 = \text{tr} (A^T A - 2CA^T B + B^T BCC^T).
\]

Since \( A \) is constant, minimizing \( J \) is equivalent to simultaneously optimizing:

\[
\max_{B,C} \text{tr} (CA^T B) \quad \text{(4.2)}
\]

\[
\min_{B,C} \text{tr} (B^T BCC^T) \quad \text{(4.3)}
\]

Note that because \( \text{tr}(XY) \leq \text{tr}(X) \text{ tr}(Y) \), minimizing Eq. 4.3 is equivalent to:

\[
\min_B \text{tr} (B^T B) \quad \text{(4.4)}
\]

\[
\min_C \text{tr} (CC^T) \quad \text{(4.5)}
\]

The KKT function of objective in eq. 4.1 is:

\[
L(B, C) = J(B, C) - \text{tr} (\Gamma_B B^T) - \text{tr} (\Gamma_C C),
\]

where \( \Gamma_B \in \mathbb{R}_{+}^{M \times K} \) and \( \Gamma_C \in \mathbb{R}_{+}^{N \times K} \) are the KKT multipliers. By applying the KKT optimality conditions to \( L \) we get:

\[
\nabla_B L = BCC^T - AC^T - \Gamma_B = 0 \quad \text{(4.6)}
\]

\[
\nabla_C L = B^T BC - B^T A - \Gamma_C^T = 0, \quad \text{(4.7)}
\]
with complementary slackness:

\[ \Gamma_B \odot B = 0, \quad \text{and} \quad \Gamma_C^T \odot C = 0, \]

where \( \odot \) denotes component-wise multiplications. Eq. 4.6 and eq. 4.7 lead to:

\[
B = (AC^T + \Gamma_B)(CC^T)^{-1} \tag{4.8}
\]

\[
C = (B^TB)^{-1}(B^TA + \Gamma_C^T). \tag{4.9}
\]

Substituting eq. 4.9 to eq. 4.2 leads to:

\[
\max_B \text{tr} \left( (B^TB)^{-1}(B^TA^T B + \Gamma_C^T A^T B) \right),
\]

which is equivalent to simultaneously optimizing:

\[
\max_B \text{tr} \left( B^T A^T B \right) \tag{4.10}
\]

\[
\max_B \text{tr} \left( \Gamma_C^T A^T B \right) \tag{4.11}
\]

\[
\min_B \text{tr} \left( B^T B \right). \tag{4.12}
\]

Similarly, substituting eq. 4.8 to eq. 4.2 leads to:

\[
\max_C \text{tr} \left( (C^T A^T C^T + C^T \Gamma_B)(CC^T)^{-1} \right),
\]

which is equivalent to simultaneously optimizing:

\[
\max_C \text{tr} \left( C^T A^T C^T \right) \tag{4.13}
\]

\[
\max_C \text{tr} \left( C^T \Gamma_B \right) \tag{4.14}
\]

\[
\min_C \text{tr} \left( CC^T \right). \tag{4.15}
\]

As shown, eq. 4.12 and eq. 4.15 recover eq. 4.4 and eq. 4.5 respectively, so there is no need to substituting eq. 4.8 and eq. 4.9 into eq. 4.3.

Now we concentrate on the basis matrix \( B \) first. Eq. 4.10 – 4.12 give alternative objectives to the original NMF objective that contain only \( B \). Note that if we consider \( A \) to be an affinity matrix induced from bipartite graph \( \mathcal{G}(A) \) (which is a reasonable thought since any feature-by-item matrix can be modeled by a bipartite graph), then \( \mathcal{G}(AA^T) \) is the feature graph where edge weights describe the similarity between corresponding vertex pairs. So, eq. 4.10 looks like ratio association applied to \( \mathcal{G}(AA^T) \). But without orthogonality constraint \( B^TB = I \) (which is the part of ratio association objective), one can optimize eq. 4.10 by setting \( B \) to an infinity matrix. However, this violates eq. 4.12 which favours small \( B \). Similarly, one can optimize eq. 4.12 by
setting $B$ to a zero matrix. But again, this violates eq. 4.10. Thus, eq. 4.10 and eq. 4.12 create implicit lower and upper bound constraints on $B$: $0 \leq B \leq \Upsilon_B$.

For convenience, eq. 4.12 can be restated as:

$$\min_B \text{tr} \left( B^T B \right) \equiv \min_B \text{tr} \left( B^T B B^T B \right).$$

(4.16)

By using the fact $\text{tr} \left( X^T X \right) = \|X\|_F^2$, eq. 4.10 can be rewritten into:

$$\min_B \left( \|B^T B\|_F^2 = \sum_i (b_i^T b_i)^2 + \sum_{i \neq j} (b_i^T b_j)^2 \right).$$

Therefore, eq. 4.10–4.12 can be restated as:

$$\max_B \text{tr} \left( B^T A A^T B \right)$$

(4.17)

$$\max_B \text{tr} \left( \Gamma^T C A^T B \right)$$

(4.18)

$$\min_B \left( \sum_i (b_{j_1}^T b_i)^2 + \sum_{i \neq j} (b_{j_2}^T b_j)^2 \right)$$

(4.19)

s.t. $0 \leq B \leq \Upsilon_B$.

Even though $B$ is now bounded, since there is no column-orthogonality constraint, maximizing eq. 4.17 can be easily done by setting each entry of $B$ to the corresponding largest possible value (in graph term this means to only create one partition on $G(AA^T)$). But this scenario results in a large value of eq. 4.19 which violates the objective. Similarly, minimizing eq. 4.19 to the smallest possible value violates eq. 4.17.

Since minimizing $j_{b_1}$ implies minimizing $j_{b_2}$, but not vice versa, simultaneously optimizing eq. 4.17 and eq. 4.19 can be done by setting $j_{b_2}$ as small as possible and balancing $j_{b_1}$ with eq. 4.17. This scenario is the relaxed ratio association applied to $G(AA^T)$, and as long as vertices in $G(AA^T)$ are clustered, it leads to the grouping of related features.

The remaining problem is eq. 4.18. Since we know nothing about $\Gamma_C$, the best bet will be making each entry of $A^T B$ as large as possible. This can be done by setting $B$ to the largest possible values, but this scenario violates eq. 4.19. So, the most reasonable scenario will be making the entries near diagonal region of $A^T B$ as large as possible. This can be achieved by using $B$ from previous discussion. As $B$ is the feature clustering indicator matrix, multiplying $A^T$ with $B$ will result in a matrix that has larger entries near diagonal region, therefore it can be expected that eq. 4.18 will have good optimality. Thus simultaneously optimizing eq. 4.17–4.19 leads to the feature clustering indicator matrix $B$. 
By applying the similar approach to the coefficient matrix $C$, optimizing eq. 4.13 - 4.15 is equivalent to optimizing:

\[
\begin{align*}
\max_C \text{tr} \left( C A^T A C^T \right) & \quad (4.20) \\
\max_C \text{tr} \left( C A^T B \right) & \quad (4.21) \\
\min_{\hat{c}} \left( \sum_i (\hat{c}_i \hat{c}_i^T)^2 + \sum_{i \neq j} (\hat{c}_i \hat{c}_j^T)^2 \right) & \quad (4.22)
\end{align*}
\]

subject to $0 \leq C \leq \Upsilon_C$,

where $\hat{c}_i$ denotes $i$-th row of $C$. By following the previous discussion on $B$, it can be shown that as long as vertices in $G(A^T A)$ are clustered, simultaneously optimizing eq. 4.20 - 4.22 leads to the item clustering indicator matrix $C$.

### 4.1. A limitation of the NMF as a clustering method.

As shown in the proof of proposition 4.1, optimizing NMF objective is equivalent to applying the relaxed ratio association to the item graph $G(A^T A)$ and the feature graph $G(A A^T)$ simultaneously. And because in the NMF, clustering membership of each point is directly determined by finding the largest projection on the axis of the decomposition rank subspace ($K$ subspace) [1], the NMF can only offer good results if the data points are linearly separable.

This is not the case with the spectral clustering, where the memberships are indirectly determined by applying k-means clustering on the resulting factors. This additional step can sometimes find correct assignments even though the data points are not linearly separable. And unfortunately, since the factors produced by the NMF are nonnegative and directly point to the cluster’s centers [1], applying k-means clustering on the factors won’t change the clustering assignments.

The following examples show the limitation of the NMF in clustering linearly inseparable data points. And for comparison, the spectral clustering is used. For the spectral clustering, we use Ng et al. algorithm (NJW) [31], and for the NMF, we use Lee & Seung algorithm (NMFLS) [15], and Kim & Park algorithm (NMFJK) [24]. NJW and NMFLS are the standard algorithm for the spectral clustering and the NMF respectively, and NMFJK is the NMF algorithm that has convergence guarantee. Algorithm 2 describes NJW algorithm, and algorithm 3 describes clustering using the NMF. Note that we wrote codes for NJW and NMFLS by ourselves, and use codes from the authors website [1] for NMFJK. To get the same treatment as in NJW, we use the same kernel strategy for NMFLS and NMFJK. The adjustable parameter $\alpha$ is learned directly from the datasets, and the results are displayed in figure 4.1, 4.2, and 4.3.

---

1. [http://www.cc.gate.edu/~jingu/nmf/index.html](http://www.cc.gate.edu/~jingu/nmf/index.html)
Algorithm 2 Spectral clustering algorithm by Ng et al. \[31\] (NJW).

1. Input: Rectangular data matrix \(A \in \mathbb{R}^{M \times N}\) with \(N\) data points, \#cluster \(K\), and Gaussian kernel parameter \(\alpha\).
2. Construct symmetric affinity matrix \(\hat{A} \in \mathbb{R}^{N \times N}\) from \(A\) by using Gaussian kernel.
3. Normalize \(\hat{A}\) by \(\hat{A} \leftarrow D^{-1/2}\hat{A}D^{-1/2}\) where \(D\) is a diagonal matrix with \(D_{ii} = \sum_j \hat{a}_{ij}\).
4. Compute the \(K\) largest eigenvectors of \(\hat{A}\), and form \(\hat{X} \in \mathbb{R}^{N \times K} = [\hat{x}_1, \ldots, \hat{x}_K]\), where \(\hat{x}_k\) is the \(k\)-th largest eigenvector of \(\hat{A}\).
5. Normalize every row of \(\hat{X}\), i.e., \(X_{ij} \leftarrow X_{ij}/(\sum_j X^2_{ij})^{1/2}\).
6. Apply k-means clustering on the row of \(\hat{X}\) to obtain the clustering indicator matrix \(\bar{X} \in \mathbb{R}^{N \times K}\).

Algorithm 3 Clustering by using the NMF.

1. Input: Rectangular data matrix \(A \in \mathbb{R}^{M \times N}\) with \(N\) data points, \#cluster \(K\), and Gaussian kernel parameter \(\alpha\).
2. Construct symmetric affinity matrix \(\hat{A} \in \mathbb{R}^{N \times N}\) from \(A\) by using Gaussian kernel.
3. Compute \(B\) and \(C\) by using NMF algorithm (NMFLS or NMFJK) so that \(\hat{A} \approx BC\).
4. Assume \(C\) is used, then clustering assignment of data point \(n, x_n\), can be computed by \(x_n \leftarrow \arg\max_k c_n, \forall n\).

As shown in figure 4.1, 4.2, and 4.3 while the spectral clustering can correctly find the clustering assignments for all datasets, the NMFs can only compete with the spectral clustering for the last dataset which is rather linearly separable. These results are in accord with the proof of proposition 4.1 (that states as long as vertices on the feature ⟨item⟩ graph are clustered, optimizing the NMF objective leads to the feature ⟨item⟩ clustering indicator matrix). Thus, it seems that as a clustering method, the NMF is more similar to k-means clustering or support vector machine (SVM) which also can only cluster linearly separable datasets, than to the spectral methods, even though both clustering using the NMF and the spectral methods are based on matrix decomposition techniques. Accordingly, clustering performances of the NMF can probably be improved by using appropriate kernel methods as in k-means clustering and SVM.

4.2. Experimental results. The experiments are conducted to evaluate the performances of the NMF as a clustering method. All algorithms are developed in GNU Octave under linux platform using a notebook with 1.86 GHz Intel processor.
and 2 GB RAM. Reuters-21578 document corpus\footnote{http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html}, the standard dataset for testing learning algorithms and other text-based processing methods, is used for this purpose. This dataset contains 21578 documents (divided into 22 files with each file contains 1000 documents and the last file contains 578 documents) with 135 topics created manually with each document is assigned to one or more topics based on its content. The dataset is available in SGML and XML format, we use the XML version. We use all but the 18\textsuperscript{th} file because this file is invalid both in its SGML and XML version. We use only documents that belong to exclusively one class (we use “classes” for refeering to the original grouping, and “clusters” for referring to groups resulted from the clustering algorithms).

Further, we remove the common English stop word\footnote{http://snowball.tartarus.org/algorithms/english/stop.txt}, stem the remaining words using Porter stemmer\footnote{[32]}, and then remove words that belong to only one document. And also, we normalize the term-by document matrix $A$ by: $A \leftarrow AD^{-1/2}$ where $D = \text{diag}(A^T A e)$ as suggested by Xu et al.\footnote{[11]}. We form test datasets by combining top 2, 4, 6, 8, 10, and 12 classes from the corpus. Table\ref{tab:statistics} summarizes the statistics of these test datasets, where #doc, #word, %nnz, max, and min refer to the number...
of document, the number of word, percentage of nonzero entry, maximum cluster size, and minimum cluster size respectively. And table 4.2 gives the sizes (#doc) of these top 12 classes.

As shown in table 4.1, a rectangular word-by-document matrix $A$, where $a_{ij}$ denotes the (weighted) frequency of word $i$ in document $j$, can be induced from each dataset (Reuters2, ..., Reuters12). Thus we have two options: either by directly applying co-clustering on bipartite graph $G(A)$ for simultaneously finding the word and document clustering, or by first transforming $G(A)$ into the corresponding unipartite graph $G(\Phi(A^T, A))$ using kernel function $\Phi$, and then applying previously discussed clustering methods (algorithm 2 and 3) for finding the document clustering. Employing kernel methods is unheard in document clustering researches (probably because of the sizes of the datasets), thus the co-clustering style will be employed instead. And actually, this is the most common way in using the NMF for clustering purpose [1, 2, 3, 4, 5, 6, 7, 16, 21, 22, 23, 24, 28, 29, 30, 33, 34].

For comparison, we will employ the spectral co-clustering on $G(A)$ which is computed by finding the first $K$ singular vectors of $A$. And because only reference classes for documents are available, we will only evaluate document clustering performances. The following theorem gives a theoretical support for the using of the SVD in mul-

![Fig. 4.3. Clustering linearly inseparable datasets using NMFJK.](image)

**Table 4.1**

Statistics of the test datasets.

| The data | #doc | #word | %nnz | max  | min  |
|----------|------|-------|------|------|------|
| Reuters2 | 6090 | 8547  | 0.363| 3874 | 2216 |
| Reuters4 | 6797 | 9900  | 0.353| 3874 | 333  |
| Reuters6 | 7354 | 10319 | 0.347| 3874 | 269  |
| Reuters8 | 7644 | 10596 | 0.340| 3874 | 144  |
| Reuters10| 7887 | 10930 | 0.336| 3874 | 114  |
| Reuters12| 8052 | 11172 | 0.333| 3874 | 75   |
Table 4.2

Sizes of the top 12 classes.

| class | #doc |
|-------|------|
| 1     | 3874 |
| 2     | 2216 |
| 3     | 374  |
| 4     | 333  |
| 5     | 288  |
| 6     | 269  |
| 7     | 146  |
| 8     | 144  |
| 9     | 129  |
| 10    | 114  |
| 11    | 90   |
| 12    | 75   |

ticlass spectral co-clustering (more detailed discussion on this topic can be found in ref. [35]), and algorithms 4 and 5 summarize the document clustering using the SVD and the NMF respectively.

Theorem 4.2. The optimal value of the following problem:

$$\max_{X^T X = Y^T Y = I_K} \text{tr}(X^T R Y), \quad (4.23)$$

is equal to $\sum_{k=1}^{K} \sigma_k$ if

$$X = [x_1, \ldots, x_K] Q, \quad \text{and} \quad Y = [y_1, \ldots, y_K] Q$$

where $R \in \mathbb{C}^{M \times N}$ denotes a full rank rectangular complex matrix with singular values $\sigma_1 \geq \ldots \geq \sigma_{\min(M,N)} > 0$, $0 \leq K \leq \min(M,N)$, $X \in \mathbb{C}^{M \times K}$ and $Y \in \mathbb{C}^{N \times K}$ denote column orthogonal matrices, $x_k$ and $y_k$ ($k \in [1, K]$) respectively denote $k$-th left and right singular vectors correspond to $\sigma_k$, and $Q \in \mathbb{C}^{K \times K}$ denotes an arbitrary unitary matrix.

Proof. Eq. (4.23) can be rewritten as:

$$\max_{X^T X = Y^T Y = I_K} \frac{1}{2} \text{tr} \left( \begin{bmatrix} X^T & Y^T \end{bmatrix} \begin{bmatrix} 0 & R \ R^T & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} \right).$$

Since $\Psi$ is a full rank Hermitian matrix, by the Ky Fan theorem (shown in theorem 4.3 below), the global optimum solution is given by the first $K$ eigenvectors of $\Psi$:

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} x_1, \ldots, x_K \\ y_1, \ldots, y_K \end{bmatrix} Q.$$

Therefore,

$$\begin{bmatrix} 0 & R \\ R^T & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \end{bmatrix} = \lambda_k \begin{bmatrix} x_k \\ y_k \end{bmatrix},$$
where $k \in [1, K]$ and $\lambda_k$ denotes $k$-th eigenvalue of $\Psi$. Then,

$$R y_k = \lambda_k x_k, \text{ and}$$

$$R^T x_k = \lambda_k y_k,$$

where $u_k$ and $y_k$ denote the left and right singular vectors associated with singular value $\lambda_k (= \sigma_k)$ of $R$. 

**Theorem 4.3** (Ky Fan [36, 37]). The optimal value of the following problem:

$$\max_{X^T X = I_K} \text{tr}(X^T H X)$$

is equal to $\sum_{k=1}^K \lambda_k$ if

$$X = [u_1, \ldots, u_K] Q,$$

where $H \in \mathbb{C}^{N \times N}$ denotes a full rank Hermitian matrix with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_N \in \mathbb{R}$, $1 \leq K \leq N$, $X \in \mathbb{C}^{N \times K}$ denotes a column orthogonal matrix, $I_K$ denotes a $K \times K$ identity matrix, $u_k \in \mathbb{C}^N$ denotes $k$-th eigenvector corresponds to $\lambda_k$, and $Q \in \mathbb{C}^{K \times K}$ denotes an arbitrary unitary matrix.

Note that, even though theoretically $\lambda_k$ can be chosen so that $\lambda_k = \sigma_k$ for all $k$, numerically $\lambda_k$ can be negative. And also, numerically $X$ & $Y$ constructed using eigenvectors of $\Psi$ can be different from using singular vectors of $R$. Therefore one should always use the SVD for computing $X$ and $Y$. And for convenience, we assume $R$ to be of full rank. The similar result can be derived for non full rank $R$.

Theorem 4.2 gives the theoretical support for directly applying graph cuts to the bipartite graph $\mathcal{G}(A)$ to get simultaneous row and column clustering or also known as (multiclass) spectral co-clustering. The following gives the objective of the multiclass spectral co-clustering:

$$\bar{X}^T \bar{X} = \bar{Y}^T \bar{Y} = I_K \text{ s.t. } \text{tr}(\bar{X}^T A \bar{Y}),$$

where $\bar{X} \in \mathbb{R}_+^{M \times K}$ and $\bar{Y} \in \mathbb{R}_+^{N \times K}$ denote the row and column clustering indicator matrices respectively. By relaxing the nonnegativity constraints, $\bar{X}$ and $\bar{Y}$ can be found by computing the first $K$ left and right singular vectors of $A$.

There are some standard metrics in evaluating clustering quality. The most commonly used metrics are **mutual information**, **entropy**, and **purity**. We will use these metrics together with an additional metric, $F$-measure. In the following, the definitions of these metrics are outlined.

**Mutual information** (MI) measures dependency between the clusters produced by the algorithms and the reference classes. The higher the MI, the more related the
Algorithm 4 Document clustering using the SVD.

1. Input: Rectangular word-by-document matrix $A \in \mathbb{R}^{M \times N}$, and #cluster $K$.
2. Normalize $A$ by: $A \leftarrow AD^{-1/2}$ where $D = \text{diag}(A^T A e)$.
3. Compute the first $K$ right singular vectors of $A$, and form $V \in \mathbb{R}^{N \times K} = [v_1, \ldots, v_K]$, where $v_k$ is the $k$-th right singular vector of $A$.
4. Apply k-means clustering on rows of $V$ to obtain document clustering indicator matrix $\bar{V} \in \mathbb{R}^{N \times K}$.

Algorithm 5 Document clustering using the NMF.

1. Input: Rectangular word-by-document matrix $A \in \mathbb{R}^{M \times N}$, and #cluster $K$.
2. Normalize $A$ by: $A \leftarrow AD^{-1/2}$ where $D = \text{diag}(A^T A e)$.
3. Compute $C$ by using NMF algorithm (NMFLS or NMFJK) so that $A \approx BC$.
4. Compute clustering assignment of $n$-th document by: $x_n \equiv \text{arg}_k \max_c c_{n,k}, \forall n$.

clusters with the classes, and therefore the better the clustering will be. It is shown that MI is a superior measure than purity and entropy [38] because it is tolerant to the difference between #cluster and #class. MI is defined with the following formula:

$$MI \equiv \sum_{r=1}^{R} \sum_{s=1}^{S} p(r, s) \log_2 \left( \frac{p(r, s)}{p(r)p(s)} \right),$$

where $r$ and $s$ denote the $r$-th cluster and $s$-th class respectively, $p(r, s)$ denotes the joint probability distribution function of the clusters and the classes, and $p(r)$ and $p(s)$ denote the marginal probability distribution functions of the clusters and the classes respectively. Note that because of inconsistency in the formulation of normalized MI (a more commonly used metric) in the literatures, we use MI instead. Accordingly, MI's values are comparable only for the same dataset.

Entropy addresses the composition of classes in a cluster. It measures uncertainty in the cluster, thus the lower the entropy, the better the clustering will be. Unlike MI, if there is discrepancy between #cluster and #class, entropy won’t be very indicative about the the clustering quality. Entropy is defined with the following:

$$\text{entropy} \equiv \frac{1}{N \log_2 S} \sum_{r=1}^{R} \sum_{s=1}^{S} c_{rs} \log_2 \frac{c_{rs}}{c_r},$$

where $N$ is the number of samples (#doc for document clustering), $c_{rs}$ denotes the
number of samples in \( r \)-th cluster that belong to \( s \)-th class, and \( c_r \) denotes the size of \( r \)-th cluster.

\( Purity \) is the most commonly used metric. It measures the percentage of the dominant class in a cluster, so the high the better. As in \( \text{entropy} \), \( \text{purity} \) is also sensitive to the discrepancy between \#cluster and \#class. \( \text{Purity} \) is defined with:

\[
\text{purity} = \frac{1}{N} \sum_{r=1}^{R} \max_s c_{rs}.
\]

And \( \text{Fmeasure} \) combines two concept in IR: \( \text{recall} \) and \( \text{precision} \). \( \text{Recall} \) measures the proportion of the retrieved relevant documents to all relevant documents, and \( \text{precision} \) measures the proportion of the retrieved relevant documents to all retrieved documents. In the context of assessing clustering quality, \( \text{Fmeasure} \) is defined with [39]:

\[
\text{F measure} \equiv \frac{1}{R} \sum_{r=1}^{R} F_r, \quad F_r = 2 \frac{\text{precision}_r \times \text{recall}_r}{\text{precision}_r + \text{recall}_r},
\]

where \( \text{precision}_r \) and \( \text{recall}_r \) denote the \( \text{precision} \) and \( \text{recall} \) of \( r \)-th cluster.

Clustering results of the SVD and the NMFs are shown in table 4.3–4.6. And time comparisons are given in table 4.7, with times for the SVD are the sum of SVD computational times and the times for performing k-means to obtain clustering assignments, and times for the NMF are simply the times for performing the NMF on the data matrices. Because we use SVD built-in function that is written in C and highly optimized, the computational times of the SVD are not really comparable to the computational times of the NMF algorithms which are written in Matlab/Octave scripts.

As shown in table 4.3 [4.6] in general, NMFJK performs as good as the SVD for all the metrics with NMFJK tends to be better for datasets with smaller \#clusters and the SVD for datasets with bigger \#clusters. Unfortunately, NMFLS which is the most popular NMF algorithm seems to only be able to give moderate results. The convergence guarantee of NMFJK can probably have some role here as converged algorithms usually can approximate the original matrices better than algorithms without convergence guarantee [17] [24].

The computational times of NMFJK seems to be promising as it is faster than NMFLS for all datasets. Note that since NMFLS and NMFJK are written in Matlab/Octave script, improving the computational performances of these algorithms is highly possible. And according to Albright et al. [40], some highly optimized NMF algorithms can be faster than SVD algorithms.
### Table 4.3

**Average mutual information over 10 trials.**

| Data   | SVD    | NMFLS  | NMFJK   |
|--------|--------|--------|---------|
| Reuters2 | 0.4951610991 | 0.4039195065 | 0.4825151487 |
| Reuters4 | 0.7345796343 | 0.6287861424 | 0.7482158671 |
| Reuters6 | 0.8367879438 | 0.7945867871 | 0.9763402437 |
| Reuters8 | 1.0342492298 | 0.9228548694 | 1.0110485952 |
| Reuters10 | 1.1754008483 | 1.0415397095 | 1.158875544 |
| Reuters12 | 1.0812313058 | 1.1325663319 | 1.2069251441 |

### Table 4.4

**Average entropy over 10 trials.**

| Data   | SVD    | NMFLS  | NMFJK   |
|--------|--------|--------|---------|
| Reuters2 | 0.4506918576 | 0.5419334502 | 0.46337808 |
| Reuters4 | 0.3491263843 | 0.4020231303 | 0.3423082679 |
| Reuters6 | 0.3675835184 | 0.3839091543 | 0.3135973194 |
| Reuters8 | 0.3185428072 | 0.3556742607 | 0.3262763521 |
| Reuters10 | 0.2957115633 | 0.330077814 | 0.3006867745 |
| Reuters12 | 0.3338525186 | 0.3195329752 | 0.2987911092 |

### Table 4.5

**Average purity over 10 trials.**

| Data   | SVD    | NMFLS  | NMFJK   |
|--------|--------|--------|---------|
| Reuters2 | 0.8623973727 | 0.821543514 | 0.8688505747 |
| Reuters4 | 0.8394880094 | 0.7941739003 | 0.8234515227 |
| Reuters6 | 0.68180582 | 0.7451047049 | 0.6042697852 |
| Reuters8 | 0.8178963893 | 0.7490580848 | 0.7780612245 |
| Reuters10 | 0.786103715 | 0.7312032458 | 0.7769747686 |
| Reuters12 | 0.6838052658 | 0.738729757 | 0.7663686041 |

### Table 4.6

**Average Fmeasure over 10 trials.**

| Data   | SVD    | NMFLS  | NMFJK   |
|--------|--------|--------|---------|
| Reuters2 | 0.8595171797 | 0.8190358778 | 0.865279454 |
| Reuters4 | 0.6255202581 | 0.5615436865 | 0.6960413891 |
| Reuters6 | 0.6487551603 | 0.4622471694 | 0.648871315 |
| Reuters8 | 0.5043941779 | 0.4040827621 | 0.4680952898 |
| Reuters10 | 0.516367264 | 0.3800132312 | 0.4842865587 |
| Reuters12 | 0.4437491506 | 0.3567059141 | 0.4333021978 |
5. LSI aspect of the NMF. This section is the second part of this paper. Here, we will first describe LSI aspect of the NMF by showing its capability in solving synonymy and polysemy problems in some synthetic datasets given that the semantic structures allow the problems to be revealed, and then evaluate this aspect more extensively by comparing the results with results of the standard LSI method—the (truncated) SVD—using real datasets.

5.1. Synonymy problems. Synonyms are different words with similar or almost similar meaning, for example \{university, college, institute\}, \{female, girl, woman\}, and \{book, novel, biography\} each is a set of synonyms. For improving recall and precision of an IR system, it is expected that the system is able to recognize the synonyms. This task is usually done by approximating the original word-by-document matrix with its truncated SVD version [9, 41].

The synthetic dataset in table 5.1 (taken from ref. [42]) shows the synonymy problems in which Mark Twain & Samuel Clemens refer to the same person, and purple & colour are closely related. As shown, Mark Twain & Samuel Clemens are not recognized as the same person; and similarly, purple & colour also are not recognized to be related. Accordingly, if a query $q$ containing \{mark, twain\} ($q^T = [1 \ 1 \ 0 \ 0 \ 0 \ 0]$) is made into the original matrix $A$, then $q^T A = [30, 0, 20, 0, 0]$. So, only Doc1 and Doc3 are retrieved, and Doc2 is lost. Similarly, if a query containing \{colour\} is made, then only Doc4 will be retrieved, and Doc5 will be lost (note that even though purple and colour can have different meanings, according to this example they are highly related, thus any query containing either one of them is expected to retrieve Doc4 and Doc5).

The synonymy problem can be resolved using LSI technique as long as there is a path that chains them together given that the path is close enough [43]. For example in table 5.1 mark & twain are connected to samuel & clemens through Doc3. So, there is a path that connects them, and it happens that the distance is close. Thus we can expect that LSI using the SVD will be able to reveal this hidden relationship.
### Table 5.1
**Dataset for describing synonymy problems.**

| Word   | Doc1 | Doc2 | Doc3 | Doc4 | Doc5 |
|--------|------|------|------|------|------|
| mark   | 15   | 0    | 0    | 0    | 0    |
| twain  | 15   | 0    | 20   | 0    | 0    |
| samuel | 0    | 10   | 5    | 0    | 0    |
| clemens| 0    | 20   | 10   | 0    | 0    |
| purple | 0    | 0    | 0    | 20   | 10   |
| colour | 0    | 0    | 0    | 15   | 0    |

### Table 5.2
**LSI using the SVD for detecting synonyms.**

| Word   | Doc1 | Doc2 | Doc3 | Doc4 | Doc5 |
|--------|------|------|------|------|------|
| mark   | 3.7  | 3.5  | 5.5  | -ε   | -ε   |
| twain  | 11   | 10   | 16   | -ε   | -ε   |
| samuel | 4.1  | 3.9  | 6.1  | -ε   | -ε   |
| clemens| 8.3  | 7.8  | 12   | -ε   | -ε   |
| purple | -ε   | -ε   | -ε   | 21   | 7.1  |
| colour | -ε   | -ε   | -ε   | 13   | 4.5  |

### Table 5.3
**LSI using the NMF (NMFLS) for detecting synonyms.**

| Word   | Doc1 | Doc2 | Doc3 | Doc4 | Doc5 |
|--------|------|------|------|------|------|
| mark   | 3.72 | 3.50 | 5.45 | 0    | 0    |
| twain  | 11.0 | 10.4 | 16.2 | 0    | 0    |
| samuel | 4.15 | 3.90 | 6.08 | 0    | 0    |
| clemens| 8.29 | 7.79 | 12.1 | 0    | 0    |
| purple | 0    | 0    | 21.0 | 7.08 |
| colour | 0    | 0    | 13.5 | 4.55 |

### Table 5.4
**LSI using the NMF (NMFJK) for detecting synonyms.**

| Word   | Doc1 | Doc2 | Doc3 | Doc4 | Doc5 |
|--------|------|------|------|------|------|
| mark   | 3.14 | 2.95 | 4.60 | 0    | 0    |
| twain  | 9.27 | 8.71 | 13.6 | 0    | 0    |
| samuel | 3.50 | 3.29 | 5.13 | 0    | 0    |
| clemens| 7.00 | 6.58 | 10.3 | 0    | 0    |
| purple | 0    | 0    | 17.3 | 5.83 |
| colour | 0    | 0    | 11.1 | 3.74 |
Similarly, colour & purple are connected through Doc4, thus LSI is also expected to be able to reveal this relationship. Table 5.2 shows the result of rank-2 matrix approximation by using the SVD to the original matrix in table 5.1 with \( \epsilon \) denotes small positive number. Note that the rank is chosen based on the number of reference classes, i.e., author names (\{mark, twain, samuel, clemens\}) and colour related terms (\{purple, colour\}). As shown, Doc1, Doc2, and Doc3 are now indexing mark, twain, samuel, and clemens. Thus, any query containing any of these words will correctly retrieve the corresponding relevant documents. And similarly, Doc4 and Doc5 are now indexing purple and colour, so any query containing at least one of these words will correctly retrieve the corresponding relevant documents.

Now, we will apply the NMF to the data matrix in table 5.1 and see whether this technique can solve the synonymy problems. Table 5.3 and 5.4 show rank-2 matrix approximations using NMFLS and NMFJK respectively. As shown, both algorithms correctly index the synonyms, and thus the NMF can also be used in solving the synonymy problems in this dataset.

5.2. Polysemy problems. LSI technique is also expected to be able to solve polysemy—word with multiple unrelated meanings—problem. By using a synthetic dataset, we will describe how the standard LSI method and the NMF solve this problem, given that polyseme presents in unrelated documents. Table 5.5 gives an example of polyseme where bank can either refers to financial institution or area near river. By inspection, it is clear that the dataset contains two different topics: financial and river, with \{Doc1, Doc3, Doc5\} & \{money, bank, interest\} are in the first topic; and \{Doc2, Doc4, Doc6\} & \{bed, river, bank\} are in the second topic. Note that the dataset is well-conditioned for describing the polysemy problem as bank presents in unrelated documents.

If a query \( \mathbf{q}_1^T = [1 0 0 1 0] \) containing \{money, bank\} is made to the original matrix \( \mathbf{A} \) in table 5.5 then \( \mathbf{q}_1^T \mathbf{A} = [2 1 2 1 1 1] \). So, only Doc1 and Doc3 are recognized as relevant, and Doc5 will not be recognized as relevant. Similarly, if a query \( \mathbf{q}_2^T = [0 0 1 1 0] \) containing \{river, bank\} is made, then \( \mathbf{q}_2^T \mathbf{A} = [1 2 1 2 1 1] \); only Doc2 and Doc4 are recognized as relevant, and Doc6 is not.

Table 5.6 shows rank-2 SVD approximation to the original matrix. If the same queries are made to the matrix \( \hat{\mathbf{A}} \) in table 5.6 then \( \mathbf{q}_1^T \hat{\mathbf{A}} = [1.86726 1.00346 1.86726 1.00346 1.40251 0.91744] \) and \( \mathbf{q}_2^T \hat{\mathbf{A}} = [1.00346 1.86726 1.00346 1.86726 0.91744 1.40251] \). Therefore, all relevant documents can be correctly retrieved, so LSI using the SVD can solve the polysemy problem in this dataset.

Now, we will see whether the NMF can also solve the problem. Table 5.7 and 5.8 show rank-2 NMF approximations using NMFLS and NMFJK respectively. Let \( \mathbf{A}_1 \) and \( \mathbf{A}_2 \) be the matrix in table 5.7 and 5.8 respectively. If the same queries are made
### Table 5.5

Dataset for describing polysemy problem.

| Word  | Doc1 | Doc2 | Doc3 | Doc4 | Doc5 | Doc6 |
|-------|------|------|------|------|------|------|
| money | 1    | 0    | 1    | 0    | 0    | 0    |
| bed   | 0    | 1    | 0    | 1    | 0    | 1    |
| river | 0    | 1    | 0    | 1    | 0    | 0    |
| bank  | 1    | 1    | 1    | 1    | 1    | 1    |
| interest | 1 | 0   | 1    | 0    | 1    | 0    |

### Table 5.6

LSI using the SVD for detecting polyseme.

| Word  | Doc1  | Doc2  | Doc3  | Doc4  | Doc5  | Doc6  |
|-------|-------|-------|-------|-------|-------|-------|
| money | 0.8082  | -0.054983  | 0.8082  | -0.054983  | 0.547139  | 0.062068  |
| bed   | -0.023949  | 1.08239  | -0.023949  | 1.08239  | 0.117052  | 0.738319  |
| river | -0.054983  | 0.80882  | -0.054983  | 0.80882  | 0.062068  | 0.547139  |
| bank  | 1.05844  | 1.05844  | 1.05844  | 1.05844  | 0.855371  | 0.855371  |
| interest | 1.08239  | -0.023949  | 1.08239  | -0.023949  | 0.738319  | 0.117052  |

### Table 5.7

LSI using the NMF (NMFLS) for detecting polyseme.

| Word  | Doc1  | Doc2  | Doc3  | Doc4  | Doc5  | Doc6  |
|-------|-------|-------|-------|-------|-------|-------|
| money | 0.801054  | 0.013549  | 0.800924  | 0.013018  | 0.558518  | 0.082122  |
| bed   | 0.013619  | 1.082496  | 0.014032  | 1.082806  | 0.098272  | 0.748645  |
| river | 0.010063  | 0.804439  | 0.010370  | 0.804670  | 0.072989  | 0.556338  |
| bank  | 1.067149  | 1.063328  | 1.067377  | 1.062928  | 0.829791  | 0.831112  |
| interest | 1.080788  | 0.018281  | 1.080612  | 0.017564  | 0.753557  | 0.110799  |

### Table 5.8

LSI using the NMF (NMFJK) for detecting polyseme.

| Word  | Doc1  | Doc2  | Doc3  | Doc4  | Doc5  | Doc6  |
|-------|-------|-------|-------|-------|-------|-------|
| money | 0.63733  | 0.040462  | 0.63733  | 0.040462  | 0.441459  | 0.098975  |
| bed   | 0.054469  | 0.858063  | 0.054469  | 0.858063  | 0.133246  | 0.594351  |
| river | 0.040458  | 0.637333  | 0.040458  | 0.637333  | 0.098975  | 0.441459  |
| bank  | 0.877087  | 0.877087  | 0.877087  | 0.877087  | 0.699338  | 0.699338  |
| interest | 0.85806  | 0.054476  | 0.85806  | 0.054476  | 0.594352  | 0.133253  |

To these matrices, then $q_1^T A_1 = [1.868 1.07688 1.8683 1.07595 1.38831 0.91323]$, $q_1^T A_2 = [1.51442 0.91755 1.51442 0.91755 1.1408 0.79831]$, $q_2^T A_1 = [1.07721 1.8677 1.07775 1.8676 0.90278 1.38745]$, and $q_2^T A_2 = [0.91754 1.51442 0.91754]$. 

---
Clustering and LSI aspects of the NMF

Table 5.9
The standard text collections in LSI.

|        | Medline | Cranfield | CISI | ADI |
|--------|---------|-----------|------|-----|
| #Doc   | 1033    | 1398      | 1460 | 82  |
| #Word  | 12011   | 6551      | 9080 | 1215|
| %NNZ   | 0.4567  | 0.85674   | 0.51701 | 2.1479 |
| #Query | 30      | 225       | 35   | 35  |

1.51442 0.79831 1.1408]. Accordingly, the polysemy problem can also be solved by using the NMF in this dataset.

5.3. Experimental results. We will now evaluate LSI aspect of the NMF by using real datasets, and compare the results with the results of the SVD. Table 7 summarizes the datasets used in the experiments where #Doc, #Word, %NNZ, and #Query denote the number of documents, the number of unique words, percentage of nonnegative entries, and the number of predefined queries made to the corresponding word-by-document matrix respectively. These datasets are the standard text collections which have been extensively used in the LSI researches.

Each of the text collections comprises of three important files. The first file contains abstracts of the documents which each indexed by a unique identifier, the second file contains the list of queries each with a unique identifier, and the third file contains a dictionary that maps every query with its manually assigned relevant documents.

The first file is the file that is used to construct the word-by-document matrix $A \in \mathbb{R}^{M \times N}_+$. To extract the unique words, the stop words and words that shorter than two characters are removed. But we do not employ any stemming and do not remove words that only belong to one document as in section 4.2. The reasons are the stemming process seems to be not popular in the LSI researches, and removing unique words in a document can potentially reduce recall since it is possible that queries contain these words. Then after $A$ is constructed, we further adjust the entry weights by using logarithmic scale, i.e., $A_{ij} \leftarrow \log(A_{ij} + 1)$, but do not normalized the columns of the matrix. This is because based on our pre-experimental results, the logarithmic scale performs better than the simple frequency of word occurrences, and normalization has a negative effect on the retrieval performances for both the SVD and the NMF for all text collections.

The second file is used to construct the query matrix $Q \in \mathbb{R}^{Q \times M}_+ = [q_1, \ldots, q_Q]^T$ where $Q$ denotes the number of queries (shown in the last row of table 5.9), $M$
denotes the number of unique words, and \( q \) denotes the \( q \)-th query vector constructed from the file. So simply by multiplying \( Q \) with the corresponding \( A \), one can get a matrix that contains scores that describe how relevant each query to the documents in the corresponding row.

And the third file is the file that maps each query to its manually assigned relevant documents. This information will be utilized as the references to measure the retrieval performances of the SVD and the NMF.

To measure the LSI performances, average precision—the standard metric in the IR researches [44] that measures \( I \)-point interpolated average pseudo-precision at recall level \([0, 1]\)—will be used. This metric captures both recall and precision concepts without inheriting the weakness from recall, i.e., perfect recall can be achieved by retrieving all documents. The following outlines the average precision definition, and more detailed discussions can be found in, e.g., ref. [42, 44, 45].

First the definition of precision will be discussed. Let \( r = q^T A \) denotes a vector that contains document scores with respect to the query vector \( q \), and let \( r \) be sorted in reverse order (larger comes first). The precision at \( n \)-th document is given by:

\[
p_n \equiv \frac{r_n}{n},
\]

where \( r_n \) denotes the number of relevant documents up to \( n \)-th position. The pseudo-precision at recall level \( x \in [0, 1] \) is defined as:

\[
\hat{p}(x) \equiv \max \{p_n \mid x \leq r_n/r_N, \ n = 1, \ldots, N\},
\]

where \( r_N \) denotes the total number of relevant documents in the collection. And \( I \)-interpolated average precision at recall level \( x \in [0, 1] \) for a single query \( q \) is defined as:

\[
\text{average precision}_q \equiv \frac{1}{I} \sum_{n=0}^{I-1} \hat{p}\left(\frac{n}{I-1}\right),
\]

where as previously defined, \( n \) denotes the \( n \)-th position in \( r \). We will use 11-point interpolated average precision \( (I = 11) \) as proposed in ref. [42] since three out of four text collections used in our experiments are similar to those used in ref. [42]. However, due to the differences in the preprocessing steps, our results won’t be similar to the results of ref. [42]. And because there are several queries in each text collection (shown in the last row of table 5.9), average precision used in this work is the average value over \#Query. So, for each text collection:

\[
\text{average precision} \equiv \frac{1}{Q} \sum_{q=1}^{Q} \text{average precision}_q,
\]
Fig. 5.1. **Average precision** values over decomposition ranks.

**Table 5.10**

| Method    | Medline  | Cranfield | CISI       | ADI        |
|-----------|----------|-----------|------------|------------|
| SVD       | 0.4967(600) | 0.3365(600) | 0.1617(170) | 0.2663(33) |
| NMFLS     | 0.4769(600) | 0.2674(600) | 0.1510(530) | **0.2674(39)** |
| NMFJK     | 0.4862(600) | 0.2871(600) | 0.1434(360) | 0.2610(40) |

where $Q$ denotes $\#Query$.

Fig. 5.1 shows the **average precision** values over decomposition ranks (for Medline, Cranfield, and CISI: $[10, 20, \ldots, 600]$, and for ADI: $[1, 2, \ldots, 40]$) for all datasets. Table 5.10 displays average values of the **average precision** over 10 trials with the values are
in format \( \text{val}(\text{rank}) \), where \( \text{val} \) denotes the average value over 10 trials at this \( \text{rank} \), and \( \text{rank} \) denotes the rank where the maximum \( \text{average precision} \) value is obtained at the first attempt (for example, in ADI at the first attempt the SVD reaches maximum \( \text{average precision} \) at rank 33, NMFLS at rank 39, and NMFJK at rank 40, and this is also the ranks where peak values are achieved in fig. 5.1 for each dataset and each method). Note that because approximate matrices produced by the SVD are unique, there is no need to repeat the computation.

As shown in fig. 5.1 in general the SVD produces better and more stable \( \text{average precision} \) values over decomposition ranks for all datasets, with tendency the many the decomposition ranks the higher the \( \text{average precision} \) values. The \( \text{average precision} \) values produced by the NMF algorithms seem to be not stable, especially for CISI. NMFJK seems to have slightly better \( \text{average precision} \) than NMFLS. This results are interesting since as discussed in section 4.2 clustering capability of NMFJK is also better than NMFLS.

While there are cases in which NMFLS and NMFJK outperform the SVD, when the computations are repeated over 10 trials and the results are averaged, as shown in table 5.10 the superiority of NMFLS and NMFJK seems to be vanished. This can be understood since NMF algorithms when converged, only stationarity of the limit points are guaranteed (so not even local-optimality is guaranteed by NMF algorithms). On the other hand, SVD algorithms not only have global-optimality guarantee, but also produce the same factors with differences only in the numerical precision [42] (at least theoretically).

Because rank-\( k \) truncated SVD can be constructed from full rank SVD by taking the first \( k \) columns of the singular matrices and the \( k \times k \) principal submatrix of the singular value matrix, the computational times for the SVD are not recorded for each decomposition rank, rather we compute full rank SVD for each dataset and record the times which are 498.59, 448.67, 237.46, and 1.1924 seconds for Medline, Cranfield, CISI, and ADI respectively. And the computational times over decomposition ranks for NMFLS and NMFJK are shown in fig. 5.2.

As shown in fig. 5.2 in general, for every dataset NMFJK is faster than NMFLS for lower ranks, but then the computational times of NMFJK are growing faster than NMFLS, resulting in slower performance for higher ranks. These results are interesting since according to the creator of NMFJK, this algorithm is the fastest NMF algorithm so far [24]. Table 4.7 can also be considered for evaluating the computational times of NMFJK which in the Reuters datasets, NMFJK is faster than NMFLS. However, since the decomposition ranks are rather very small (up to 12), the results in 4.7 are in accord to the results in fig. 5.2. Thus, it seems that in lower ranks, NMFJK is faster than NMFLS, but in higher ranks NMFJK is slower than NMFLS. Table 5.11 shows the average computational times of NMFJK and NMFLS over 10
Fig. 5.2. Computational times over decomposition ranks (second).

Table 5.11
Average computational times over 10 trials.

|       | Medline | Cranfield | CISI  | ADI    |
|-------|---------|-----------|-------|--------|
| NMFLS | 367.8   | 283.5     | 158.6 | 0.4165 |
| NMFJK | 746.4   | 537.8     | 259.9 | 0.5045 |

trials for decomposition ranks shown in table 5.10. As the ranks are all high, NMFJK is slower than NMFLS for all datasets.

6. Conclusions. We have presented a theoretical framework for supporting clustering aspect of the NMF without setting the KKT multipliers to zeros. Thus the
stationary point used in proving this aspect is guaranteed to be on the nonnegative orthant which is the feasible region of the NMF. Our theoretical work implies a limitation of the NMF as a clustering method in which it cannot be used in clustering linearly inseparable datasets. So, the NMF as a clustering method is more resembling k-means clustering or SVM than the spectral clustering, even though both the NMF and the spectral methods utilize matrix decomposition techniques. As the clustering capabilities of k-means and SVM usually can be improved by using the kernel methods, probably the same approach can also be employed in the NMF. We will address this issue in our future researches.

Clustering capability of NMFJK is comparable to the SVD in Reuters datasets with NMFJK tends to be better for small #cluster and the SVD for big #cluster. But unfortunately, NMFLS which is the standard NMF algorithm cannot outperform the SVD. These results imply clustering aspect of the NMF is algorithm-dependent, a fact that seems to be overlooked in the NMF researches.

LSI aspect of the NMF seems to be comparable to the SVD in its power for solving synonymy and polysemy problems for datasets with clear semantic structures that allowed these problems to be revealed. In real datasets, however, the NMF generally cannot outperform the SVD. But an interesting fact comes into sight; in some cases, the NMF can outperform the SVD, even though when the computations are repeated and averaged over the number of trials, these advantages vanish. Because the NMF can offer different results depending on the algorithms, the initializations, the objectives, and the problems, improving LSI capability of the NMF is possible. We will address this problem in our future researches.

REFERENCES

[1] W. Xu, X. Liu and Y. Gong, “Document clustering based on non-negative matrix factorization,” Proc. ACM SIGIR, pp. 267-73, 2003.
[2] F. Shahnaz, M.W. Berry, V. Pauca, and R.J. Plemmons, “Document clustering using non-negative matrix factorization,” Information Processing & Management, 42(2), pp. 267-73, 2003.
[3] H. Kim and H. Park, “Sparse non-negative matrix factorizations via alternating non-negativity constrained least squares for microarray data analysis,” Bioinformatics, Vol. 23(12), pp. 1495-502, 2007.
[4] D. Lee and H. Seung, “Learning the parts of objects by non-negative matrix factorization,” Nature, 401(6755), pp. 788-91, 1999.
[5] P.O. Hoyer, “Non-negative matrix factorization with sparseness constraints,” The Journal of Machine Learning Research, Vol. 5, pp. 1457-69, 2004.
[6] V.P. Pauca, J. Piper, and R.J. Plemmons, “Nonnegative matrix factorization for spectral data analysis,” Linear Algebra and Its Applications, Vol. 416(1), pp. 29-47, 2006.
[7] C. Ding, X. He, and H.D. Simon, “On the equivalence of nonnegative matrix factorization and spectral clustering,” Proc. SIAM Data Mining Conference, pp. 606-10, 2005.
[8] J. Kim and H. Park, “Sparse nonnegative matrix factorization for clustering,” CSE Technical
Clustering and LSI aspects of the NMF

[9] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” Journal of the American Society for Information Science, 41(6), pp. 391-407, 1990.

[10] E. Gaussier and C. Goutte, “Relation between PLSA and NMF and implications,” Proc. 28th ACM SIGIR, pp. 601-2, 2005.

[11] C. Ding, T. Li, and W. Peng, “On the equivalence between Non-negative Matrix Factorization and Probabilistic Latent Semantic Indexing,” Computational Statistics & Data Analysis, Vol. 52(8), pp. 3913-27, 2008.

[12] P. Paatero and U. Tapper, “Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values,” Environmetrics, 5, pp. 111-26, 1994.

[13] P. Anttila, P. Paatero, and U. Tapper, “Source identification of bulk wet deposition in Finland by positive matrix factorization,” Atmospheric Environment, 29(14), pp. 1705-15, 1995.

[14] I.S. Dhillon and S. Sra, “Generalized nonnegative matrix approximation with bregman divergences,” UTCS Technical Reports, The University of Texas at Austin, 2005.

[15] D. Lee and H. Seung, “Algorithms for non-negative matrix factorization,” Proc. Advances in Neural Processing Information Systems, pp. 556-62, 2000.

[16] S.Z. Li, X.W. Hou, H.J. Zhang, and Q.S. Cheng, “Learning spatially localized, parts-based representation,” Proc. IEEE Comp. Soc. Conf. on Computer Vision and Pattern Recognition, pp. 207-12, 2001.

[17] C.J. Lin, “On the convergence of multiplicative update algorithms for nonnegative matrix factorization,” IEEE Transactions on Neural Networks, Vol. 18(6), pp. 1589-96, 2007.

[18] C.J. Lin, “Projected gradient methods for non-negative matrix factorization,” Technical Report ISTTech-95-013, Department of CS, National Taiwan University, 2005.

[19] H. Hindi, “A tutorial on convex optimization,” Proc. American Control Conference, pp. 3252-65, 2004.

[20] L. Grippo and M. Sciandrone, “On the convergence of the block nonlinear Gauss-Seidel method under convex constraints,” Operation Research Letters, Vol. 26, pp. 127-36, 2000.

[21] D. Kim, S. Sra, and I.S. Dhillon, “Fast projection-based methods for the least squares nonnegative matrix approximation problem,” Stat. Anal. Data Min., Vol. 1(1), pp. 38-51, 2008.

[22] D. Kim, S. Sra, I.S. Dhillon, “Fast newton-type methods for the least squares nonnegative matrix approximation problem,” Proc. SIAM Conference on Data Mining, pp. 343-54, 2007.

[23] H. Kim and H. Park, “Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method,” SIAM. J. Matrix Anal. & Appl., Vol. 30(2), pp. 713-30, 2008.

[24] J. Kim and H. Park, “Toward faster nonnegative matrix factorization: A new algorithm and comparisons,” Proc. 8th IEEE International Conference on Data Mining, pp. 353-62, 2008.

[25] M.H.V. Benthem and M.R. Keenan, “Fast algorithm for the solution of large-scale nonnegativity-constrained least squares problems,” Journal of Chemometrics, Vol. 18, pp. 441-50, 2004.

[26] R. Bro and S.D. Jong, “A fast non-negativity-constrained least squares algorithm,” Journal of Chemometrics, Vol. 11, pp. 393-401, 1997.

[27] C.L. Lawson and R.J. Hanson, “Solving least squares problems,” SIAM Classic in Applied Mathematics, 1995.

[28] T. Li and C. Ding, “The relationships among various nonnegative matrix factorization methods for clustering,” Proc. ACM 6th Int’l Conf. on Data Mining, pp. 362-71, 2006.

[29] C. Ding, T. Li, and M.I. Jordan, “Convex and semi-nonnegative matrix factorizations,” IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 45-55, 2010.

[30] C. Ding, T. Li, W. Peng, and H. Park, “Orthogonal nonnegative matrix t-factorizations for clus-
tering,” Proc. 12th ACM SIGKDD Int’l Conf. on Knowledge Discovery and Data Mining, pp. 126-35, 2006.
[31] A.Y. Ng, M.I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” Proc. 14th Advances in Neural Information Processing Systems, pp. 849-56, 2001.
[32] S.E. Robertson, C.J. van Rijsbergen, and M.F. Porter, “New models in probabilistic information retrieval,” Technical Report, British Library Research and Development, 1980.
[33] D. Cai, X. He, X. Wu, and J. Han, “Non-negative matrix factorization on manifold,” Proc. 8th IEEE International Conference on Data Mining, pp. 63-72, 2008.
[34] Q. Gu and J. Zhou, “Co-clustering on manifolds,” Proc. 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 359-68, 2009.
[35] A. Mirzal, “Clustering and Latent Semantic Indexing Aspects of the Singular Value Decomposition,” http://arxiv.org/abs/1011.4104.
[36] I. Nakić and K. Veselić, “Wielandt and ky-fan theorem for matrix pairs,” Linear Algebra and its Applications, Vol. 369(17), pp. 77-93, 2003.
[37] H. Zha, X. He, C. Ding, H. Simon, and M. Gu, “Spectral relaxation for k-means clustering,” Proc. 14th Advances in Neural Information Processing Systems, pp. 1057-64, 2001.
[38] A. Strehl and J. Ghosh, “Cluster ensembles – a knowledge reuse framework for combining multiple partitions,” Journal of Machine Learning Research, Vol. 3, pp. 583-617, 2002.
[39] N.O. Andrews and E.A. Fox, “Recent developments in document clustering,” Technical Report VA 24060, Department of CS, Virginia Tech., 2007.
[40] R. Albright, J. Cox, D. Duling, A. Langville, and C. Meyer, “Algorithms, Initializations, and Convergence for the Nonnegative Matrix Factorization,” NCSU Technical Report Math 81706, North Carolina State University, 2006.
[41] M.W. Berry, S.T. Dumais, and G.W. O’Brien, “Using linear algebra for intelligent information retrieval,” SIAM Rev., Vol. 37(4), pp. 573-95, 1995.
[42] T.G. Kolda and D.P. O’Leary, “A semidiscrete matrix decomposition for latent semantic indexing information retrieval,” ACM Trans. Inf. Syst., Vol. 16(4), pp. 322-46, 1998.
[43] A. Kontostathis and W.M. Pottenger, “A framework for understanding latent semantic indexing (LSI) performance,” Information Processing and Management, Vol. 42(1), pp. 56-73, 2006.
[44] D.K. Harman, “Overview of the Third Text Retrieval Conference (TREC-3),” National Institute of Standards and Technology, 1996.
[45] M.W. Berry and M. Browne, “Understanding Search Engines: Mathematical Modelling and Text Retrieval, 2nd edition,” SIAM, Philadelphia, 2005.