DC-NMF: NONNEGATIVE MATRIX FACTORIZATION BASED ON DIVIDE-AND-CONQUER FOR FAST CLUSTERING AND TOPIC MODELING

RUNDONG DU[†], DA KUANG[‡], BARRY DRAKE[§¶], AND HAESUN PARK[¶]

Abstract. The importance of unsupervised clustering and topic modeling is well recognized with ever-increasing volumes of text data available from numerous sources. Nonnegative matrix factorization (NMF) has proven to be a successful method for cluster and topic discovery in unlabeled data sets. In this paper, we propose a fast algorithm for computing NMF using a divide-and-conquer strategy, called **DC-NMF**. Given an input matrix where the columns represent data items, we build a binary tree structure of the data items using a recently-proposed efficient algorithm for computing rank-2 NMF, and then gather information from the tree to initialize the rank-k NMF, which needs only a few iterations to reach a desired solution. We also investigate various criteria for selecting the node to split when growing the tree.

We demonstrate the scalability of our algorithm for computing general rank-k NMF as well as its effectiveness in clustering and topic modeling for large-scale text data sets, by comparing it to other frequently utilized state-of-the-art algorithms. The value of the proposed approach lies in the highly efficient and accurate method for initializing rank-k NMF and the scalability achieved from the divide-and-conquer approach of the algorithm and properties of rank-2 NMF. In summary, we present efficient tools for analyzing large-scale data sets, and techniques that can be generalized to many other data analytics problem domains along with an open-source software library called *SmallK*.

Key words. Constrained low rank approximation, nonnegative matrix factorization, divide and conquer, clustering, topic modeling, text analysis, scalable algorithms

AMS subject classifications. 15A23, 62H30, 65F30, 65K05, 90C26, 90C30, 90C59

1. Introduction. [1]

Matrix low rank approximations [17] have played a crucial role as one of the most fundamental tools in machine learning, data mining, image processing, information retrieval, computer vision, signal processing, and other areas of computational science and engineering. Low rank approximations based on the singular value decomposition (SVD) or Principal Component Analysis (PCA) provide a compact representation of a large matrix. This compact representation not only enables data compression and dimension reduction, but also allows for the discovery of latent structures in high-dimensional, large volumes of data. Throughout this paper, we will assume that a data set is represented in a matrix $A \in \mathbb{R}^{m \times n}$ where m is the number of features and the columns of A represent the n data items. Assuming that $\text{rank}(A) = r$ and k is a positive integer $k \leq r$, a lower rank matrix $\hat{A}$ with $\text{rank}(\hat{A}) = k$, which is closest to A in matrix L_2 norm or Frobenius norm, can be obtained from the SVD of A [17]. A general objective function for a low rank approximation problem

[†]School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332-0160 (rdu@gatech.edu).

[‡]Department of Mathematics, University of California, Los Angeles, CA 90095-1555 (dakuang@math.ucla.edu).

[§]Georgia Tech Research Institute, Georgia Institute of Technology, Atlanta, GA 30318 (barry.drake@gtri.gatech.edu).

[¶]School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0765 (hpark@cc.gatech.edu).

[1]The new algorithm DC-NMF introduced in this paper is based on the fast rank-2 NMF and hierarchical NMF algorithms presented in [31]. However, the two papers are substantially different. Some of the key differences and the new contributions of this paper are summarized towards the end of this section.

can be expressed as

$$\min_{W,H} \|A - WH\|_F. \tag{1.1}$$

Although the above objective function is not convex, the SVD gives us a global optimal solution $W \in \mathbb{R}^{m\times k}$ and $H \in \mathbb{R}^{k\times n}$ when there is no constraint on the factors W and H. The columns of W form a basis for the space spanned by the columns of A and the ith column of H can be viewed as a k-dimensional representation of the ith column of A in the space spanned by W. With $k \leq \text{rank}(A)$, the columns of W (and the rows of H) are expected to be linearly independent as the solution for Eqn. (1.1) should provide the best approximation for A.

Many key problems in data analytics can be formulated by adding constraints to (1.1). With constraints on W and H, the approximation error $\|A - WH\|_F$ will increase from that given by the SVD. However, the ability to formulate a specific data analytics problem using the additional constraints makes the low rank approximation result more useful in practice in spite of possibly larger approximation error. One of the best known examples of constrained matrix low rank approximation is the nonnegative matrix factorization (NMF). Compared to the SVD solution, by imposing nonnegativity constraints on the low rank factors W and H, interpretability is achieved when the data originate from the nonnegative domain. Examples of this are images, chemical concentrations, and, more recently, large volumes of text data. Although NMF can be defined for any matrix, we assume that a data set is represented in a nonnegative matrix $A \in \mathbb{R}_+^{m\times n}$, where $\mathbb{R}_+$ is the set of nonnegative real numbers.

The research on NMF since its introduction [44, 34] has continued to explode and NMF has established its importance in numerous domains in spite of its much shorter history compared to the SVD [17]. For a review of NMF, please see [27, 19]. Numerous algorithms have been designed for NMF, and NMF has been successfully applied to clustering, topic modeling, and other real-life applications such as cancer subtype detection [21], blind source separation for audio [43], and many others. However, algorithms for computing NMF are more expensive than those for the SVD. In particular, assuming the size of the input matrix A is fixed, the computational time tends to increase superlinearly for typical NMF algorithms as we discover finer structures with larger values for k in a data set. Our proposed NMF algorithm based on divide-and-conquer addresses this issue.

The contributions of this paper are as follows. First, we present a highly scalable fast algorithm called DC-NMF (Divide-and-Conquer NMF) for computing the standard NMF. In [31], we introduced an algorithm called HierNMF2, for hierarchical topic modeling via a fast rank-2 NMF and a binary tree splitting rule for text data. For DC-NMF, we utilize the leaf nodes of the cluster tree computed from HierNMF2 to obtain a highly informed initial guess for the matrix W in Eqn. (2.1) and, with a few subsequent NLS (Nonnegativity-constrained Least Squares) iterations, obtain the standard NMF very fast. We also introduce new methods to recursively increase the reduced rank. These methods do not depend on the data type while the method introduced in [31] was specifically designed for topic modeling of document data. Second, we illustrate that DC-NMF provides a very fast, scalable, and accurate method for clustering and topic modeling of all data sizes. Some of the key advantages of the NMF-based formulation are that, unlike many other algorithms based on statistical foundations such as LDA [6], we can analyze the behavior of the algorithm more readily, utilize highly optimized existing matrix computation routines for solving the subproblems efficiently, and accordingly produce scalable and effective algorithms for

large-scale problems. Some of the recent work on NMF algorithms assume separability constraints [1, 4, 33], but our experiments show that these constraints are too restrictive in high noise, real-world problems as discussed and illustrated in Section 5. Lastly, we present substantial experimental comparisons on real text data sets that validate the efficiency and the quality of DC-NMF results. We also provide case studies and meaningful and easily explorable data graphs that DC-NMF generates on a real text corpus. DC-NMF is implemented in both Matlab and an open source C++ package called `SmallK` [7].

The rest of the paper is organized as follows. In Section 2, we describe a unified framework for clustering and topic modeling (document clustering) using NMF. In Section 3, we discuss the computational advantages of rank-2 NMF over rank-k NMF and review our efficient algorithm for rank-2 NMF. In Section 4, we introduce our divide-and-conquer strategy and several node splitting criteria. In Section 5, we empirically compare our method with existing state-of-the-art methods. In Section 6, we conclude our paper and discuss its implication on other work.

2. NMF for Clustering and Topic Modeling. The characteristics that distinguish NMF from other matrix approximation methods such as the SVD are the nonnegativity constraints on W and H. We can interpret these lower rank matrices more easily within the context of many application domains. In fact, one can view clustering as a special case of dimension reduction: in a clustered data set, the cluster representatives can be viewed as the basis vectors, and each data item can be represented as a linear combination of these cluster representative vectors. Clustering has been an essential tool in the analysis of large scale data sets. Clustering appear under various names in different contexts such as hard clustering, soft clustering, [46, 8, 25, 32], and topic modeling for text data [6, 5, 20, 31], and they refer to fundamental tools with a similar goal: to organize the data sets into coherent groups where between-group relationships are remote and the data items within a group are closely related. We will sometimes refer to all these methods as "clustering" in a very broad sense. Topic modeling is an unsupervised method that discovers topics in a text collection along with the keywords. For soft clustering of text data, the cluster representative vectors reveal the topics. Accordingly, for text data sets, we may view soft clustering and topic modeling as equivalent.

Consider a factorization of a matrix $A = [\mathbf{a}_1, \cdots, \mathbf{a}_n] \in \mathbb{R}_+^{m\times n}$, $A = WH$, where $W \in \mathbb{R}_+^{m\times k}$ and $H \in \mathbb{R}_+^{k\times n}$, and $k \leq \min(m, n)$. Suppose $\|\mathbf{a}_i\|_1 = 1$ $(1 \leq i \leq n)$. If $\|\mathbf{w}_i\|_1 = 1$, we can interpret the columns of W as probability distributions over features. In the case of text data, a column of W with this sum-to-one constraint can be viewed as a topic, which is a probability distribution over keywords. Hard clustering can be interpreted as: the ith data item belongs to the jth cluster when j is the index of the largest entry in the ith column of H. With this view, hard clustering, soft clustering, and topic modeling can be unified in the same model where W reveals the cluster representatives or topics and H contains the cluster membership weights. In reality, especially when k is small, the equality $A = WH$ will not hold and we need to consider an approximation [2, 1, 33]. Many problems in data analytics can be formulated using NMF with various difference measures such as matrix norms or Bregman divergences. In this paper, we will focus on the most commonly utilized NMF formulation based on the Frobenius norm

$$\min_{W\geq 0, H\geq 0} \|A - WH\|_F. \tag{2.1}$$

Main advantages of the Frobenius norm based NMF are its flexibility for design-

ing efficient and scalable algorithms for large-scale problems and its ability to produce more accurate solutions in a variety of noisy real-life applications even when other measures such as KL-divergence can model the problems better theoretially [25, 28, 31, 32, 15, 37]. NMF as a topic modeling method has several advantages over LDA (Latent Dirichlet Allocation), a probabilistic and generative model-based method. First, we can provide a term-document matrix with *tf-idf* weighting as an input to NMF instead of raw frequencies of word occurrences, as in most text classification methods [40]. Tf-idf weighting has been widely shown to improve classification and clustering accuracy. Second, numerous matrix computation and optimization routines that have been studied and implemented with high computational efficiency and rigorous analysis can provide a basis for efficiently computing the solutions of NMF [39, 26, 30, 10], making NMF-based methods highly scalable for web-scale topic modeling.

A considerable number of papers on clustering have been devoted to K-means, and K-means remains as one of the most popular clustering methods [24]. The K-means objective function can be expressed as

$$\min_{H\in\{0,1\}^{k\times n},\mathbf{1}_k^T H=\mathbf{1}_n^T,W} \|A-WH\|_F, \tag{2.2}$$

where $\mathbf{1}_k \in \mathbf{R}^{k\times 1}, \mathbf{1}_n \in \mathbf{R}^{n\times 1}$ are vectors with "1"'s at all the entries. Although K-means and NMF are related, due to the difference in the constraints they may produce very different clustering results [32, 25, 28],

3. Low Rank Approximation with Reduced Rank 2. The NMF algorithm, DC-NMF, that we propose in this paper relies on certain properties of NMF when $k = 2$, which makes applying divide-and-conquer possible. In this section, we offer some theoretical and algorithmic justifications for DC-NMF. We discuss the relationships between SVD and NMF, the quality of approximation by NMF and by SVD when $k = 2$, and review a very fast algorithm for rank-2 NMF introduced in [31].

3.1. Rank-2 Approximation by SVD and NMF. Although the SVD and NMF differ only by the nonnegativity constraints as shown in Eqns. (1.1) and (2.1), there is still much to study regarding the relationships between SVD and NMF. The rank and nonnegative rank of a matrix can be defined using a single framework: For a $A \in \mathbb{R}^{m\times n}$, $rank(A)$ is the smallest integer p for which there exist $U \in \mathbb{R}^{m\times p}$ and $V \in \mathbb{R}^{p\times n}$ such that $A = UV$. The nonnegative rank, $rank_+(A)$, is the smallest integer p for which there exist $U \in {\mathbb{R}_+}^{m\times p}$ and $V \in {\mathbb{R}_+}^{p\times n}$ such that $A = UV$. One notable result is that for a nonnegative matrix A, NMF is the same as SVD when $k = 1$ due to the well known Perron-Frobenius Theorem [22]. It follows from the theorem that there are nonnegative left and right singular vectors associated with the leading singular value of $A \in \mathbb{R}_+^{m\times n}$.

When $k > 1$, the low rank approximations by NMF and SVD are not the same in general. Clearly, the approximation error by NMF cannot be smaller than that of the SVD since NMF imposes more constraints. The nonnegative rank of a matrix $A \in \mathbb{R}_+^{m\times n}$ is the same as the smallest possible number of vertices of a convex hull that contain all columns of A when projected onto the $(m-1)$-dimensional simplex [9]. This relationship has an important implication for rank-2 NMF, as illustrated below. Cohen and Rothblum [11] showed an interesting relationship between SVD and NMF: given $A \in \mathbb{R}_+^{m\times n}$, if $\text{rank}(A) = 2$, then $\text{rank}_+(A) = 2$. They also provided a constructive method to generate rank-2 NMF when $\text{rank}(A) = 2$. Without loss of generality, we can assume that $\|A(:,i)\|_1 = 1$ for every column of A. Under this

FIG. 3.1. *Illustration of NMF of rank-2 and rank-3 matrices. (a) When a nonnegative matrix has rank 2, its columns can be projected onto a one-dimensional subspace, and we can find two extreme endpoints (the red dots in this figure) which define a convex hull that encloses all the points. (b) The columns of a matrix* $A \in \mathbb{R}_+^{4\times 4}$ *with rank 3 (four solid dots in the figure) are projected onto a two-dimensional subspace. The depicted region is the intersection of the 3-dimensional simplex with this two-dimensional subspace, with four sides on the boundary. There are no three vertexes that define a convex hull that encloses all the four points, and thus* $\text{rank}_+(A) > 3$.

TABLE 3.1

Relative difference in the approximation errors produced by SVD and NMF for sparse matrices of various sizes with 1% non-zero entries uniformly distributed in the interval $(0, 1)$. *For each pair of* (m, n), *the results were the average over 100 random matrices* $A \in \mathbb{R}_+^{m\times n}$. *The approximation error of NMF was computed by taking the minimum of 20 random runs for each matrix.*

$k = 2$

n \ m	300	500	1000	3000
250	0.7704×10^{-4}	1.3296×10^{-4}	1.8039×10^{-4}	1.6177×10^{-4}
300	1.0103×10^{-4}	1.4303×10^{-4}	1.7583×10^{-4}	1.6051×10^{-4}

$k = 3$

n \ m	300	500	1000	3000
250	2.0238×10^{-4}	2.9706×10^{-4}	3.5395×10^{-4}	3.4395×10^{-4}
300	2.4668×10^{-4}	3.0484×10^{-4}	3.6593×10^{-4}	3.3824×10^{-4}

assumption, when $k = 2$, all columns of A lie on a one-dimensional simplex; therefore, there must exist two columns of A that define two extreme rays of a 2-d nonnegative cone which encloses all the columns (Fig. 3.1(a)). When $\text{rank}(A) > 2$, this property does not hold in general. For example, when $\text{rank}(A) = 3$, under the sum-to-one constraint, the columns of A lie on a two-dimensional subspace, and there is not always a convex hull with three vertices that encloses all columns of A (Fig. 3.1(b)).

Although the solution of rank-2 NMF can be computed based on the constructive proof for a theorem provided in [11], its usage is limited to the case with a nonnegative matrix A with $\text{rank}(A) = 2$. When $\text{rank}(A) > 2$, one may consider computing its rank-2 approximation by SVD $\hat{A} = U_2\Sigma_{2\times 2}V_2^T$ first. One difficulty is that $\hat{A}$ will not necessarily be nonnegative although $\text{rank}(\hat{A}) = 2$. On the other hand, simply setting the negative elements in $\hat{A}$ to zero will change its rank.

We have observed empirically that when the reduced rank $k = 2$, the relative difference between the approximation errors by SVD and NMF were very small. The difference was measured by $|(error_{\text{nmf}} - error_{\text{svd}})/error_{\text{svd}}|$ (Table 3.1), where $error_{\text{nmf}}$ and $error_{\text{svd}}$ are the approximation errors by NMF and SVD, respectively. We also noticed that this relative difference becomes larger as k increases.

3.2. A Fast Algorithm for Rank-2 NMF. In [27], it was shown that many of the promising NMF algorithms can be explained using the block coordinate descent (BCD) framework [39, 26, 30, 3, 18]. A natural application of the BCD framework to NMF partitions the unknowns into two blocks, elements in W and H, and iteratively solves alternating multiple right-hand side NLS (ANLS) problems for each of W and H until some stopping criterion is satisfied.

The ANLS algorithms for NMF differ in the way the NLS subproblems are solved. An NLS problem with a single right-hand side vector $\min_{\mathbf{g}\geq 0} \|B\mathbf{g} - \mathbf{y}\|_F$ can be solved by active-set-type algorithms [30]. In active-set-type algorithms, we identify a

Algorithm 1 Algorithm for solving $\min_{G\geq 0}\|BG-Y\|_F$, where $B=[\mathbf{b}_1,\mathbf{b}_2]\in\mathbb{R}_+^{m\times 2}, Y\in\mathbb{R}_+^{m\times n}$

1: Solve unconstrained least squares $G^{\emptyset}=[\mathbf{g}_1^{\emptyset},\cdots,\mathbf{g}_n^{\emptyset}]\leftarrow\min\|BG-Y\|$
2: $\beta_1\leftarrow\|\mathbf{b}_1\|$, $\beta_2\leftarrow\|\mathbf{b}_2\|$, $\mathbf{u}\leftarrow(Y^T\mathbf{b}_1)/\beta_1^2$, $\mathbf{v}\leftarrow(Y^T\mathbf{b}_2)/\beta_2^2$
3: **for** $i=1$ to n
4: **if** $\mathbf{g}_i^{\emptyset}\geq 0$ **then return** $\mathbf{g}_i^{\emptyset}$
5: **else**
6: **if** $u_i\beta_1\geq v_i\beta_2$ **then return** $[u_i,0]^T$
7: **else return** $[0,v_i]^T$
8: **end if**
9: **end if**
10: **end for**

partitioning of variables in $\mathbf{g}$ into $\mathbf{g}_{\mathcal{A}}$ and $\mathbf{g}_{\mathcal{P}}$, whose indices are in an *active set* $\mathcal{A}$ and a *passive set* $\mathcal{P}$, respectively. The optimal passive set is the one where the solution of unconstrained least squares is feasible [13], i.e., $\mathbf{g}_{\mathcal{P}}>0$, *and* $\|B\mathbf{g}-\mathbf{y}\|_2^2$ is minimized. Because the number of possible active sets is exponential in k, a well-guided search of the optimal active sets is important, such as presented by the active-set and block principle pivoting methods [26, 45].

We now summarize some of the key features of the fast NMF algorithm for $k=2$ that we introduced in [31]. When $k=2$, $J(\mathbf{g})\stackrel{\text{def}}{=}\|B\mathbf{g}-\mathbf{y}\|_2^2=\|\mathbf{b}_1g_1+\mathbf{b}_2g_2-\mathbf{y}\|_2^2$, where $B=[\mathbf{b}_1,\mathbf{b}_2]\in\mathbb{R}_+^{m\times 2}$, $\mathbf{y}\in\mathbb{R}_+^{m\times 1}$, and $\mathbf{g}=[g_1,g_2]^T\in\mathbb{R}^{2\times 1}$, and the number of possible active sets is reduced to $2^2=4$. Unlike in a standard iterative optimization algorithm such as the projected gradient descent (PGD) method where the algorithm strucure is not directly affected by the value of k, in an active-set method, when $k=2$, we can directly and effectively obtain the optimal active set by choosing the one with the smallest $J(\mathbf{g})$ among all the feasible solutions $\mathbf{g}\geq 0$ as follows. If the solution $\mathbf{g}^{\emptyset}=argmin\|B\mathbf{g}-\mathbf{y}\|_2$ for the unconstrained problem is nonnegative, then $\mathbf{g}^{\emptyset}$ is the solution for the nonnegativity constrained problem. Otherwise, between the solutions for the two unconstrained problems $\min\|\mathbf{b}_ig_i-\mathbf{y}\|_2$ $(i=1,2)$, which are always feasible since $\mathbf{b}_i\geq 0$ and $\mathbf{y}\geq 0$, we can efficiently choose the best one. We exclude $\mathbf{g}=(0,0)^T$ since one of the above three is always better. For NLS with multiple right-hand sides, it is not cache-efficient to compute the solutions for the above three cases separately. Better computational efficiency emerges in our algorithm when we solve NLS with n right hand side vectors $\mathbf{y}_i$ simultaneously, which is summarized in Algorithm 1. The entire for-loop (lines 3-10, Algorithm 1) is embarrassingly parallel and can be vectorized. To achieve this, unconstrained solutions for all three possible passive sets are computed before entering the for-loop. Algorithm 1 represents a non-random pattern of memory access, and is much faster for Rank-2 NMF than applying existing active-set-type algorithms directly.

4. Fast NMF based on Divide-and-Counquer. In this section, we propose a fast algorithm for computing NMF for any given $k\geq 2$, which we call DC-NMF (Divide-and-Conquer NMF). Based on the fast rank-2 NMF algorithm and a divide-and-conquer method, DC-NMF computes a high quality W for NMF, which we show in the following subsection. We will also provide and compare several alternative formulations for DC-NMF.

The value of k represents the number of clusters or number of topics, which is often larger than 2. In addition, since a larger k value produces a better low rank

approximation, a fast algorithm that works for $k > 2$ is needed. Increasing the reduced rank k in the unconstrained low rank approximation (1.1) strictly improves the approximation quality until k reaches $\text{rank}(A)$ [17]. For NMF, the following similar result holds.

THEOREM 4.1. *For $A \in \mathbb{R}_+^{m\times n}$ with $\text{rank}_+(A) = k$, we have*

$$\min_{W^{(p+1)}\geq 0, H^{(p+1)}\geq 0} \|A - W^{(p+1)}H^{(p+1)}\|_F < \min_{W^{(p)}\geq 0, H^{(p)}\geq 0} \|A - W^{(p)}H^{(p)}\|_F$$

for all $p < k$, where $W^{(p)} \in \mathbb{R}_+^{m\times p}$ and $H^{(p)} \in \mathbb{R}_+^{p\times n}$.

Proof. Let $(W_*^{(p)}, H_*^{(p)}) = \arg\min_{W^{(p)}\geq 0, H^{(p)}\geq 0} \|A - W^{(p)}H^{(p)}\|_F$, and $R^{(p)} = (r_{ij})_{m\times n} = A - W_*^{(p)}H_*^{(p)}$. For $p < k$, we have $R^{(p)} \neq 0$, and we can prove at least one element of $R^{(p)}$ is positive. Assume $r_{ij} > 0$ for some i, j. Then we have

$$\|R^{(p+1)}\|_F \leq \|A - \begin{bmatrix} W_*^{(p)} & r_{ij}\mathbf{e}_i^m \end{bmatrix} \begin{bmatrix} H_*^{(p)} \\ (\mathbf{e}_j^n)^T \end{bmatrix} \|_F < \|R^{(p)}\|_F$$

where e_i^m (e_i^n) is the i-th unit vector in $\mathbb{R}^m$ ($\mathbb{R}^n$).

Now we prove that $R^{(p)}$ has at least one positive element. Assume $R^{(p)} \neq 0$ and has no positive element. Then any nonzero column, say jth column, of $R^{(p)}$: $R_{:j}^{(p)} = (r_{1j}, \ldots, r_{mj})^T \neq 0$ has at least one negative element. Let's choose the greatest negative component, say $r_{ij} < 0$. Since $a_{i,j} \geq 0$, $r_{ij} < 0$, and

$$R_{:j}^{(p)} = [r_{1j}, \ldots, r_{mj}]^T = [a_{1j}, \ldots, a_{mj}]^T - \sum_{l=1}^{p} [w_{1l}, \ldots, w_{ml}]^T h_{lj} \leq 0,$$

there always exists an index $\hat{l}$, such that $h_{\hat{l}j} \neq 0$ and $w_{i\hat{l}} \neq 0$. Then we can choose a small enough $\epsilon > 0$ and replace $h_{\hat{l}j}$ by $\tilde{h}_{\hat{l}j} = h_{\hat{l}j} - \epsilon > 0$ such that

$$\left\|[\tilde{r}_{1j}, \ldots, \tilde{r}_{mj}]^T\right\|_F = \left\|[r_{1j}, \ldots, r_{mj}]^T + \epsilon[w_{1\hat{l}}, \ldots, w_{m\hat{l}}]^T\right\|_F < \left\|[r_{1j}, \ldots, r_{mj}]^T\right\|_F.$$

However, this contradicts the assumption that $R^{(p)}$ is minimized. ∎

The above theorem shows that in the context of NMF for a nonnegative matrix, the approximation error is strictly reduced when the reduced rank k is increased, until k reaches the nonnegative rank.

4.1. Proposed Algorithm: DC-NMF. The NMF problem shown in (2.1) can be recast as

$$\min_{W\geq 0}\min_{H\geq 0} \|A - WH\|_F^2 = \min_{W\geq 0} \left\{ \min_{\mathbf{x}_1,\ldots,\mathbf{x}_n \in \text{span}_+(W)} \|A - [\mathbf{x}_1, \ldots, \mathbf{x}_n]\|_F^2 \right\} \stackrel{\text{def}}{=} \min_{W\geq 0} e_A(W)$$

where $\text{span}_+(W) \stackrel{\text{def}}{=} \{\Sigma_{i=1}^k h_i\mathbf{w}_i | h_i \in \mathbb{R}_+\}$ is the conical hull of $W = [\mathbf{w}_1, \ldots, \mathbf{w}_k] \in \mathbb{R}_+^{m\times k}$. Accordingly, rank-$k$ NMF can be interpreted as finding a nonnegative basis $\mathbf{w}_1, \ldots, \mathbf{w}_k$ such that A can be best approximated by vectors in $\text{span}_+(\mathbf{w}_1, \ldots, \mathbf{w}_k)$. We will use the notation $A \approx \text{span}_+(\mathbf{w}_1, \ldots, \mathbf{w}_k)$ to denote that A is approximated by the conical hull of nonnegative vectors $\mathbf{w}_i$'s.

Unlike for the SVD, one cannot use successive rank-1 deflations to go from rank-2 NMF to rank-k NMF for $k > 2$ [27]. For NMF, all vectors in $W \in \mathbf{R}^{m\times k}$ typically change completely when the reduced rank k changes. However, since rank-2 NMF can be used for binary clustering, the columns of A can be divided into two clusters based on H from rank-2 NMF, forming two submatrices A_1 and A_2, as illustrated in Figure 4.1. Assume we have a rank-2 NMF of A as $A \approx \text{span}_+(\mathbf{w}_1, \mathbf{w}_2)$. Then we view $\mathbf{w}_i$ as a representative vector for A_i, i.e., $A_i \approx \text{span}_+(\mathbf{w}_i)$, for $i = 1, 2$. If $\text{rank}_+(A) > 2$, then we can obtain a better approximation of A by replacing one

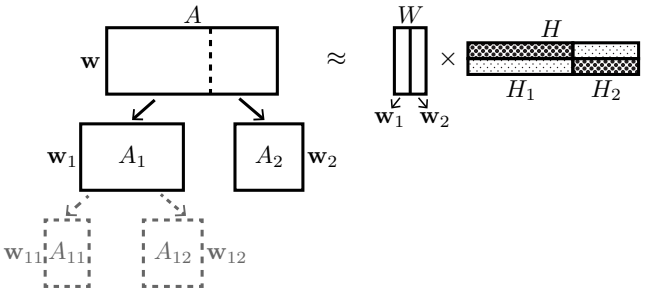

FIG. 4.1. *Illustration of how DC-NMF use divide-and-conquer to go from rank-2 NMF to higher rank NMF. The dark part in H means relative larger values.*

of $\mathbf{w}_1$ and $\mathbf{w}_2$ with two basis vectors obtained by applying rank-2 NMF on A_1 or A_2. Our cluster tree traversing rule determines this next submatrix for which we increase the reduced rank for NMF approximation from 1 to 2, so that the overall nonnegative approximation error for A is locally reduced the most. For example, in Figure 4.1, applying rank-2 NMF on A_1 gives us two new submatrices A_{11} and A_{12} where $A_{11} \approx \text{span}_+(\mathbf{w}_{11})$ and $A_{12} \approx \text{span}_+(\mathbf{w}_{12})$. Then according to Theorem 4.1, we have $e_{A_1}([\mathbf{w}_{11}, \mathbf{w}_{12}]) < e_{A_1}(\mathbf{w}_1)$. Since the total approximation error for A is controlled by such local errors (see Theorem 4.2), A is better approximated by vectors in $\text{span}_+(\mathbf{w}_{11}, \mathbf{w}_{12}, \mathbf{w}_2)$, which gives us a good rank-3 approximation. We repeat the above divide-and-conquer steps until we reach the desired reduced rank k. The above procedure consists of the following three key components:

S1. We partition a matrix A into two submatrices A_1 and A_2 according to the factor H from the rank-2 NMF of A by the following rule: the j-th column of A belongs to A_1 if $H[1, j] > H[2, j]$; otherwise it belongs to A_2. We then take $\mathbf{w}_i$ as a representative vector for A_i, $A_i \approx \text{span}_+(\mathbf{w}_i)$, $i = 1, 2$. We denote this procedure as $(A_1, A_2, \mathbf{w}_1, \mathbf{w}_2) = \text{SPLIT}(A)$.

S2. Suppose matrix $A \approx \text{span}_+(\mathbf{w})$ in step S1. We measure the effect of the reduced approximation error from the increase of the reduced rank from 1 to 2 for A by the score $e_A(\mathbf{w}) - (e_{A_1}(\mathbf{w}_1) + e_{A_2}(\mathbf{w}_2)) = \min_{\mathbf{h}} \|A - \mathbf{w}\mathbf{h}^T\|_F^2 - \sum_{i=1}^2 \min_{\mathbf{h}_i} \|A_i - \mathbf{w}_i\mathbf{h}_i^T\|_F^2$. We denote this procedure as $score = \text{COMPUTE\_SCORE}(A, A_1, A_2, \mathbf{w}, \mathbf{w}_1, \mathbf{w}_2)$. Note that the solutions $\mathbf{h}$ and $\mathbf{h}_i$ will be automatically nonnegative since A, A_i, $\mathbf{w}$, $\mathbf{w}_i$ are all nonnegative.

S3. We recursively apply Step **S1** $(k-1)$ times, dividing columns of A into k clusters and obtaining one representative vector for each cluster, resulting in k vectors in total, which are the column vectors of the desired $W \in \mathbb{R}_+^{m \times k}$. Each time we apply **S1**, we choose to further split the submatrix that will result in the largest approximation error decrease measured by the local score defined in **S2**. This step is described in detail in Algorithm DC-W. In Algorithm DC-W, each A_i is represented by $\mathbf{w}_i$ and we use $W = [\mathbf{w}_1, \ldots, \mathbf{w}_k]$ to represent A.

THEOREM 4.2. *Suppose $A \in \mathbb{R}_+^{m \times n}$, and the columns of A are partitioned into $[A_1, \ldots, A_k]$ with $2 \le k \le n$, and $W = [\mathbf{w}_1, \ldots, \mathbf{w}_k] \in \mathbb{R}_+^{m \times n}$. Then $e_A(W) \le \sum_{i=1}^k e_{A_i}(\mathbf{w}_i)$ i.e. $\min_{H \in \mathbb{R}_+^{k \times n}} \|A - WH\|_F^2 \le \sum_{i=1}^k \min_{\mathbf{h}_i \in \mathbb{R}_+^{n \times 1}} \|A_i - \mathbf{w}_i\mathbf{h}_i^T\|_F^2$.*

Proof. Let $\widehat{\mathbf{h}}_i = \arg\min_{\mathbf{h}} \|A_i - \mathbf{w}_i\mathbf{h}^T\|_F^2$ and $\widehat{H} = [\widehat{H}_1, \ldots, \widehat{H}_k]$, where $\widehat{H}_i = [0, \ldots, 0, \widehat{\mathbf{h}}_i, 0, \ldots, 0]^T$. Then, $\min_H \|A - WH\|_F^2 \le \|A - W\widehat{H}\|_F^2 = \sum_{i=1}^k \|A_i - W\widehat{H}_i\|_F^2 =$

Algorithm DC-W Algorithm to generate basis vectors $W = [\mathbf{w}_1, \ldots, \mathbf{w}_k] \in \mathbb{R}_+^{m\times k}$ for an input matrix $A \in \mathbb{R}_+^{m\times n}$ and reduced dimension $k > 2$ based on Rank-2 NMF and tree-traversing rule.

$A_1 \leftarrow A$, $\text{score}(A_1) \leftarrow \infty$
$(A_{11}, A_{12}, \mathbf{w}_{11}, \mathbf{w}_{12}) \leftarrow \text{SPLIT}(A_1)$
for $l = 2 : k$
  $j \leftarrow \arg\max_{1\le i<l} \text{score}(A_i)$
  $A_j \leftarrow A_{j1}$, $\mathbf{w}_j \leftarrow \mathbf{w}_{j1}$, $A_l \leftarrow A_{j2}$, $\mathbf{w}_l \leftarrow \mathbf{w}_{j2}$
  if $l < k$ **then**
    $(A_{j1}, A_{j2}, \mathbf{w}_{j1}, \mathbf{w}_{j2}) \leftarrow \text{SPLIT}(A_j)$
    $\text{score}(A_j) \leftarrow \text{COMPUTE\_SCORE}(A_j, A_{j1}, A_{j2}, \mathbf{w}_j, \mathbf{w}_{j1}, \mathbf{w}_{j2})$
    $(A_{l1}, A_{l2}, \mathbf{w}_{l1}, \mathbf{w}_{l2}) \leftarrow \text{SPLIT}(A_l)$
    $\text{score}(A_l) \leftarrow \text{COMPUTE\_SCORE}(A_l, A_{l1}, A_{l2}, \mathbf{w}_l, \mathbf{w}_{l1}, \mathbf{w}_{l2})$
  end if
end for

$$\sum_{i=1}^{k} \|A_i - \mathbf{w}_i\widehat{\mathbf{h}}_i^T\|_F^2 = \sum_{i=1}^{k} \min_{\mathbf{h}_i} \|A_i - \mathbf{w}_i\mathbf{h}_i^T\|_F^2. \ \square$$

The matrix A (after a proper permutation of columns), the partition $A_1, \ldots, A_k$ and the representative vectors $\mathbf{w}_1, \ldots, \mathbf{w}_k$ from Algorithm DC-W satisfy the conditions in Theorem 4.2. This means that if we use the W collected from Algorithm DC-W and obtain H from one step of NLS $\min_{H\ge 0} \|WH - A\|_F$ to get W and H as the NMF solution, the approximation error $\|A - WH\|_F^2$ will be bounded by $\sum_{i=1}^{k} e_{A_i}(\mathbf{w}_i)$, which is what we minimize in each step of Algorithm DC-W. We can also perform several more NLS iterations for NMF to further reduce the approximation error using W from Algorithm DC-W as the initial guess for W. The stopping criteria for flat NMF can be used here, for example the one used in [30] that checks whether the solution is a stationary point. In practice, we have found that there is usually a significant drop of approximation error after one full alternating iteration of computing H and then updating W, and subsequent iterations did not significantly reduce the approximation error. Therefore, in our proposed DC-NMF, we perform one iteration to compute H and update W starting with W given by Algorithm DC-W, in order to obtain a good solution while maintaining the speed advantage. The approximation error $\|A - WH\|_F^2$ can be computed by the formula $\|A-WH\|_F^2 = \|A\|_F^2 - 2\cdot\text{trace}(HA^TW) + \text{trace}(W^TWHH^T)$ to avoid directly computing $A - WH$, which is computationally expensive and can destroy the sparse structure of A.

4.2. Other Possibilities for DC-NMF. The priority scores for DC-NMF proposed in [31, 15] need to pre-split a cluster (of columns) in order to compute a priority score. We can also define heuristic scores that do not need a pre-split. For example, supposing $\tilde{A}$ is a submatrix corresponding to a cluster and $\tilde{\mathbf{w}}$ is its representative vector, we can define a heuristic score as $\min_{\tilde{\mathbf{h}}} \|\tilde{A} - \tilde{\mathbf{w}}\tilde{\mathbf{h}}^T\|_F$ to check how well $\tilde{A}$ is represented as rank-1 matrix $\tilde{\mathbf{w}}\tilde{\mathbf{h}}^T$ i.e., how coherent its columns are, and split (i.e. approximate by rank 2) the worst represented cluster.

To describe these priority scores in a unified way, we use the notations as shown in Fig. 4.1, with tilde added to each symbol, such that $\tilde{A} \approx \text{span}_+(\tilde{\mathbf{w}})$, $\tilde{A}_i \approx \text{span}_+(\tilde{\mathbf{w}}_\mathbf{i})$ $(i = 1, 2)$, where $\tilde{A}$ is a submatrix of the original data matrix A, consisting of a cluster

TABLE 4.1
Various priority scores for choosing a cluster to split.

Name	Formula	Need Pre-split	Note
Score 0	$s = \min_{\tilde{\mathbf{h}}} \|\tilde{A} - \tilde{\mathbf{w}}\tilde{\mathbf{h}}^T\|_F^2 - \sum_{i=1}^{2} \min_{\tilde{\mathbf{h}}_i} \|\tilde{A}_i - \tilde{\mathbf{w}}_i\tilde{\mathbf{h}}_i^T\|_F^2$	Y	The score proposed in this paper
Score 1	$s = \min_{\mathbf{u},\mathbf{v}} \|\tilde{A} - \mathbf{u}\mathbf{v}^T\|_F^2 - \sum_{i=1}^{2} \min_{\mathbf{u}_i,\mathbf{v}_i} \|\tilde{A}_i - \mathbf{u}_i\mathbf{v}_i^T\|_F^2$	Y	The score used for hierarchical clustering in [15]
Score 2	$s = \text{mNDCG}(\tilde{\mathbf{w}}_1) \times \text{mNDCG}(\tilde{\mathbf{w}}_2)$	Y	The score used for hierarchical topic modeling in [31]
Score 3	$s_i = \min_{\tilde{\mathbf{h}}} \|\tilde{A}_i - \tilde{\mathbf{w}}_i\tilde{\mathbf{h}}^T\|_F^2$	N	Measures how well $\tilde{A}_i$ is represented by $\tilde{\mathbf{w}}_i$.
Score 4	$s_i = \min_{\mathbf{u},\mathbf{v}} \|\tilde{A}_i - \mathbf{u}\mathbf{v}^T\|_F^2$	N	Measures how close $\tilde{A}_i$ is to a rank-1 matrix.
Score 5	$s_i = \|\tilde{A}_i - \tilde{W}\tilde{H}_i\|_F^2$	N	Measures how well $\tilde{A}_i$ is represented by $\tilde{W}$.

of columns of A. After one step of rank-2 NMF, the matrix $\tilde{A}$ is divided into two submatrices $\tilde{A}_1$ and $\tilde{A}_2$. For methods that do not need pre-split, we can directly compute priority score s_1, s_2 for $\tilde{A}_1$ and $\tilde{A}_2$, using $\tilde{\mathbf{w}}_1$ and $\tilde{\mathbf{w}}_2$, respectively. However, for methods that need pre-split, we can only compute the priority score s for $\tilde{A}$ with the same information. We summarize some of the priority scores in Table 4.1, where $\tilde{\mathbf{h}}$, $\mathbf{u}$ and $\mathbf{v}$ are column vectors of proper size.

In our experiments, we found that Score 0 and Score 1 often obtain significantly lower approximation errors than the other scores. However, Score 1 requires significantly longer computation time than Score 0 since Score 1 also computes a rank-1 SVD. Our tests show that when we start two DC-NMF computations with Score 0 and Score 1 at the same time, by the time DC-NMF with Score 1 completes computation of W from Algorithm DC-W, DC-NMF with Score 0 completes computation of an initial W and runs several alternating NLS iterations for NMF, obtaining better solutions than DC-NMF with Score 1. Therefore, we recommend Score 0 in practice.

5. Experiments. In this section, we show experimental results for DC-NMF and compare it with state-of-the-art algorithms for NMF, clustering, and topic modeling. First, we focus on the role of DC-NMF as a generic algorithm for computing NMF and evaluate its runtime versus approximation error. Then, we apply DC-NMF to small- to medium-scale data sets with ground-truth to evaluate its effectiveness for clustering before moving to much larger data sets for the benchmarking of computational efficiency. Our experiments were run on a server with two Intel E5-2620 processors, each having six cores, and 377 GB memory.

Before proceeding to the experimental results, we first describe the data sets and experimental settings in detail.

5.1. Data Sets. Six text data sets were used in our experiments: 1. **Reuters-21578**[2] contains news articles from the Reuters newswire in 1987. We discarded documents with multiple class labels, and then selected the 20 largest classes. 2. **20 Newsgroups**[2] contains articles from Usenet newsgroups and have a defined hierarchy of 3 levels. Usenet users post messages and reply to posts under various discussion

[2] `http://www.daviddlewis.com/resources/testcollections/reuters21578/` (retrieved in June 2014)

[2] `http://qwone.com/~jason/20Newsgroups/` (retrieved in June 2014)

TABLE 5.1
Data sets used in our experiments. Numbers in parentheses are the numbers of clusters/topics we requested for unlabeled data sets.

Data sets	Has label	Has hierarchy	# terms	# docs	# nodes at each level
Reuters-21578	Y	N	12,411	7,984	20
20 Newsgroups	Y	Y	36,568	18,221	6/18/20
Cora	Y	N	154,134	29,169	70
NIPS	Y	N	17,981	447	13
RCV1	N	-	149,113	764,751	(60)
Wiki-4.5M	N	-	2,361,566	4,126,013	(80)

boards, often including a personalized signature at the end of their messages. Unlike the widely-used indexing of this data set[2], we observed that many articles had duplicate paragraphs due to cross-referencing. We discarded cited paragraphs and signatures, which increased the difficulty of clustering. 3. **Cora** [41] is a collection of research papers in computer science, from which we extracted the title, abstract, and reference-contexts. Although this data set comes with a predefined topic hierarchy of 3 levels, we observed that some topics, such as "AI – NLP" and "IR – Extraction", were closely related but resided in different subtrees. Thus, we ignored the hierarchy and obtained 70 ground-truth classes as a flat partitioning. 4. **NIPS** is a collection of NIPS conference papers. We chose 447 papers from the 2001-2003 period [16], which were associated with labels indicating the technical area (algorithms, learning theory, vision science, etc). 5. **RCV1** [36] is a much larger collection of news articles from Reuters, containing about 800,000 articles from the time period of 1996-1997. We used the entire collection as an unlabeled data set. 6. **Wikipedia**[4] is an online, user-contributed encyclopedia and provides periodic dumps of the entire website. We processed the dump of all the English Wikipedia articles from March 2014, and used the resulting 4.5 million documents as an unlabeled data set **Wiki-4.5M**, ignoring user-defined categories.

We summarize these data sets in Table 5.1. The first four medium-scale data sets have ground-truth labels for the evaluation of cluster quality, while the remaining two large scale data sets are treated as unlabeled. All the labeled data sets except 20 Newsgroups have very unbalanced sizes of ground-truth classes. We constructed the normalized-cut weighted version of term-document matrices as in [46].

5.2. Implementation. We implemented DC-NMF both in Matlab and in an open-source C++ software library called `SmallK`[5] [7]. The existing methods we compared DC-NMF with are grouped into three categories: NMF algorithms, clustering methods and topic modeling methods. Though clustering and topic modeling can be unified in the framework of matrix factorization, as explained in Section 1, we label a method as belonging to one of the two categories according to the task for which it was originally targeted.

NMF Algorithms. We compared the following algorithms for computing rank-k NMF:[6]

- `MU`: The multiplicative update algorithm for Frobenius-norm based NMF [35]. `MU` is

[4] `https://dumps.Wikimedia.org/enWiki/`

[5] `https://smallk.github.io/`

[6] Besides the listed algorithms, we also experimented with a recent algorithm based on coordinate descent with a greedy rule to select the variable to improve at each step [23]. However, this algorithm became increasingly slow when we increased k and kept the size of A the same. Therefore, we did not include it in our final comparison.

not guaranteed to converge to a stationary point solution although it reduces the objective function after each iteration.
- `ANLS/BPP`: The block principal pivoting algorithm that follows the two-block coordinate descent framework [29, 30]. We will often refer to this method as simply BPP.
- `HALS/RRI`: The hierarchical alternating least squares algorithm [10, 19], which is a $2k$-block coordinate descent method. We will simply refer to this as `HALS`.

Many schemes that can be used to accelerate the above algorithms have been proposed in the literature (e.g. [38, 42]) but our comparisons will be on the above baseline algorithms.

Clustering Methods. The clustering methods we compared include:
- `nmf-hier`: Hierarchical clustering based on standard NMF with ANLS and an active-set method for NLS [26]. The active-set method searches through the space of active-set/passive-set partitionings for the optimal active set, with a strategy that reduces the objective function at each search step.
- `nmf-flat`: Flat clustering based on standard NMF with ANLS. The block principal pivoting (BPP) method [29, 30] is used as an exemplar algorithm to solve the NLS subproblems. In our experiments, multiplicative update rule algorithms [34] were always slower and gave similar quality compared to active-set-type algorithms, thus were not included in our results.
- `kmeans-hier`: Hierarchical clustering based on standard K-means. We used the hierarchical clustering workflow described in [31].
- `kmeans-flat`: Flat clustering based on standard K-means.
- `CLUTO`: A clustering toolkit[7] written in C++. We used the default method in its `vcluster` program, namely a repeated bisection algorithm.

Topic Modeling Methods. The topic modeling methods we compared include:
- `Mallet-LDA`: The software MALLET[8] written in Java for flat topic modeling, which uses the Gibbs sampling algorithm for LDA. 1000 iterations were used by default.
- `AnchorRecovery`: A recent fast algorithm to solve NMF with separability constraints [1]. It selects an "anchor" word for each topic, for example, "Los Angeles Clippers" rather than "basketball", which could carry a narrow meaning and not semantically represent the topic [1]. The software is written in Java[9]. We used the default parameters.
- `XRAY`: Another recent algorithm to solve NMF with separability constraints [33]. It incrementally selects "extreme rays" to find a cone that contains all the data points. We used the *greedy* option as the selection criteria in this algorithm.
- `Hottopixx`: A recent method that formulates Separable NMF as a linear program and solves it using incremental gradient descent [4]. We used the default parameters.

5.3. Experimental Settings. To evaluate the cluster and topic quality, we use the *normalized mutual information* (NMI). Here we use the computed cluster membership labels as the input symbol sets [12]. NMI is a measure of the similarity between two flat partitionings, and is only applicable to data sets for which the ground-truth labels are known. It is particularly useful when the number of generated clusters is different from that of ground-truth labels and can be used to determine the optimal number of clusters. More details can be found in [40]. For data sets with defined

[7] `http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview`
[8] `http://mallet.cs.umass.edu/`
[9] `https://github.com/mimno/anchor`

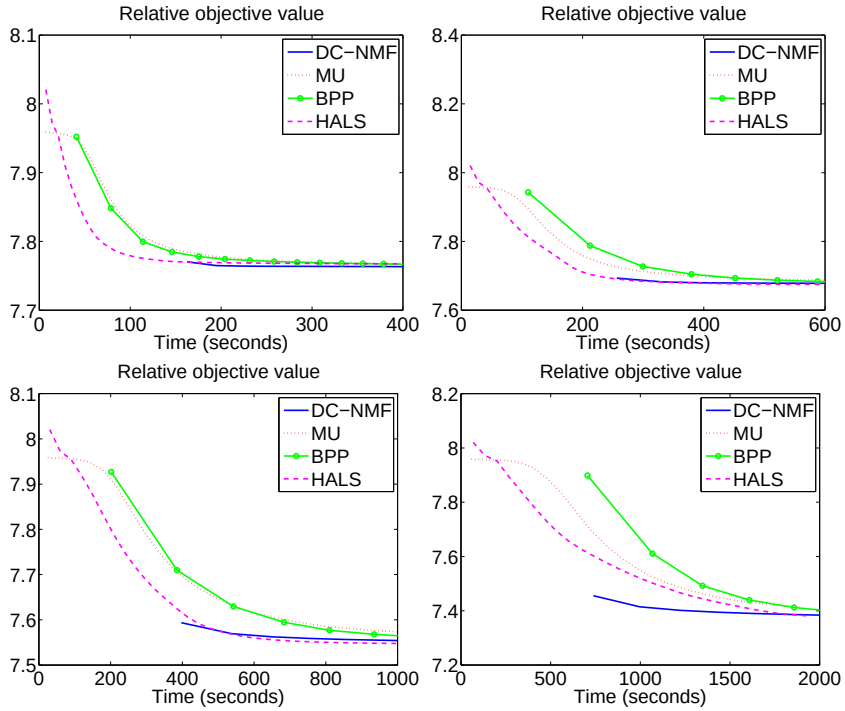

FIG. 5.1. *Comparison of approximation error between DC-NMF versus other algorithms for computing NMF. Results are shown for* $k = 20, 40, 80.160$.

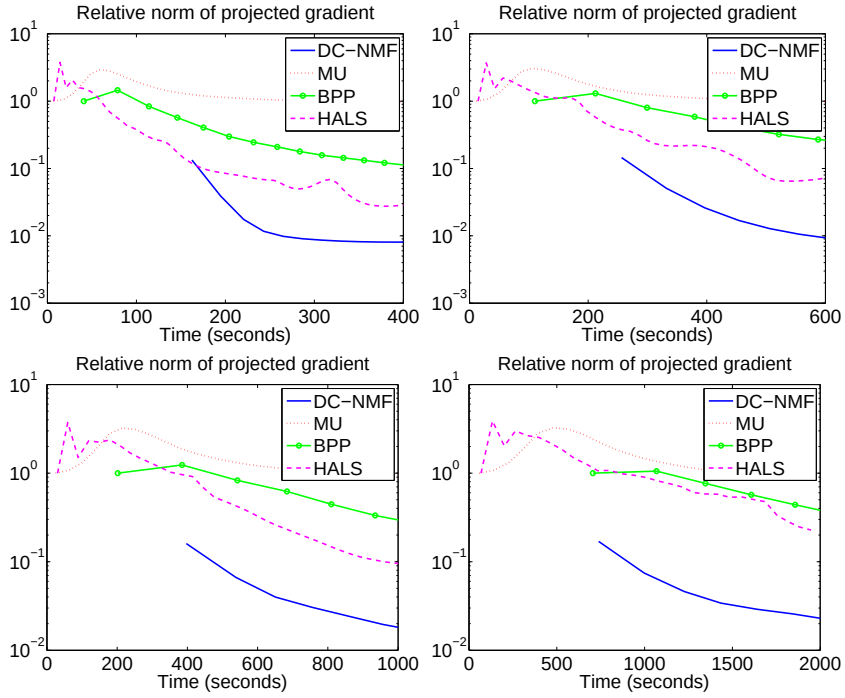

FIG. 5.2. *Comparison of projected gradient norm between DC-NMF versus other algorithms for computing NMF. Results are shown for* $k = 20, 40, 80.160$.

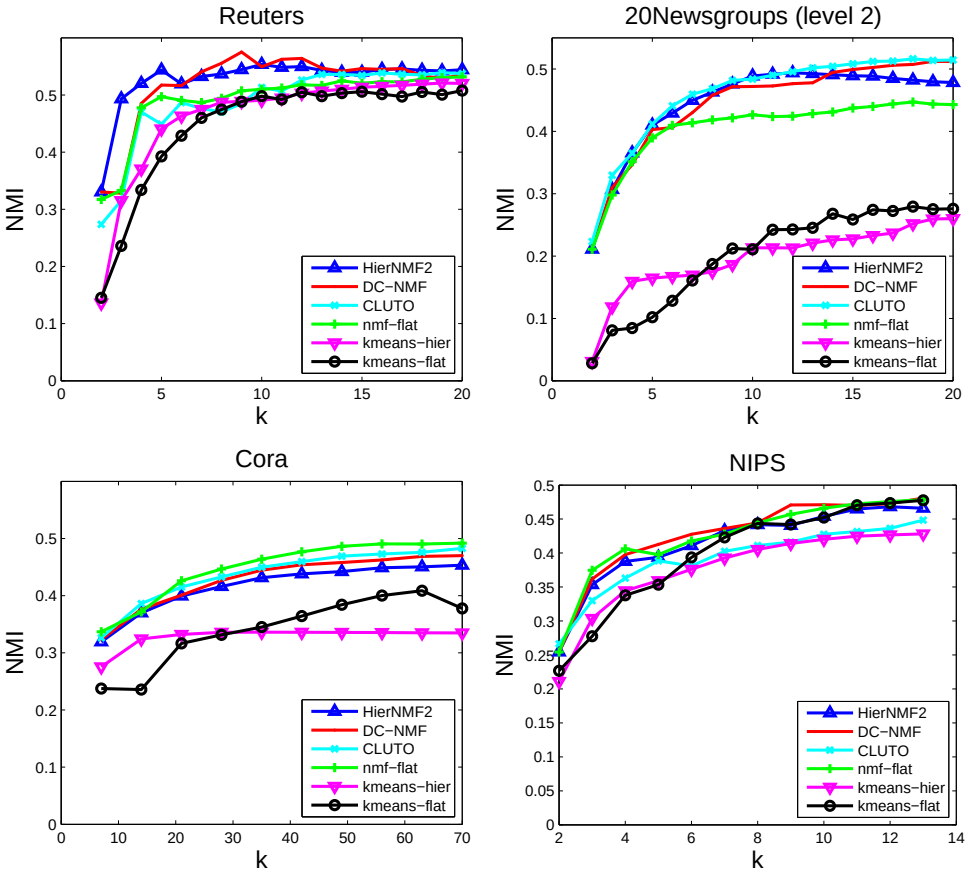

FIG. 5.3. *DC-NMF versus other* clustering methods *in cluster quality evaluated by normalized mutual information (NMI).*

hierarchy, we compute NMI between a generated partitioning and the ground-truth classes at each level of the ground-truth tree. In other words, if the ground-truth tree has depth L, we compute L NMI measures, one for each level. When evaluating the results given by DC-NMF (Algorithm DC-W), we treat all the outliers as one separate cluster for fair evaluation.

Hierarchical clusters and flat clusters cannot be compared against each other directly. When evaluating the hierarchical clusters, we take snapshots of the tree as leaf nodes are generated, and treat all the leaf nodes in each snapshot as a flat partitioning which is to be compared against the ground-truth classes. This is possible since the leaf nodes are non-overlapping. Thus, if the maximum number of leaf nodes is set to c, we produce $c-1$ flat partitionings forming a hierarchy. For each method, we perform 20 runs with random initializations. Average measurements are reported. Note that for flat clustering methods, each run consists of $c-1$ separate executions with the number of clusters set to $2, 3, \cdots, c$.

The maximum number of leaf nodes c is set to be the number of ground-truth labels at the deepest level for labeled data sets (see Table 5.1); and we set $c = 60$ for **RCV1** and $c = 80$ for **Wiki-4.5M**. The Matlab `kmeans` function has a batch update phase and a more time consuming online update phase. We rewrote this function

using BLAS-3 operations and boosted its efficiency substantially[10]. We use both phases for data sets with fewer than 20,000 documents, and only the batch-update phase for data sets with more than 20,000 documents. For NMF, we use the projected gradient norm as the stopping criterion [39] with a tolerance parameter $\epsilon = 10^{-4}$. The projected gradient norm is sensitive to the scaling of the W and H factors: WD and $D^{-1}H$ yield the same approximation error but different values of projected gradient norm, where D is a diagonal matrix with positive entries on the diagonal (see details in [30, 14]). To ensure a fair comparison between different methods, before computing a projected gradient norm, we make the columns of W have unit 2-norm and scale H accordingly. All the methods are implemented with multi-threading.

5.4. DC-NMF for Computing Rank-k NMF. Since our focus in this paper is on large-scale NMF, we compare the algorithms for computing rank-k NMF on a large-scale text data set, namely RCV1. We report the approximation error and projected gradient norm achieved by each algorithm in Fig. 5.1 and Fig. 5.2. While the approximation error measures the effectiveness of an NMF algorithm, the projected gradient norm determines when to stop the algorithm, and is thus important for the run-time in actual use of these algorithms. The results show that HALS and DC-NMF produce the smallest approximation error and DC-NMF has more advantages when k increases. We observed that DC-NMF achieves much smaller projected gradient norm than the other methods and is faster. Note that HALS may run into problems of divide-by-zero and we also found that the results were very sensitive to the way zeros were treated numerically. DC-NMF algorithm does not have such a problem nor require any parameters.

5.5. DC-NMF for Clustering and Topic Modeling.

5.5.1. Cluster Quality. Figs. 5.3 and 5.4 show the cluster quality on four labeled data sets, comparing DC-NMF with the state-of-the-art *clustering methods* and *topic modeling methods*, respectively. `nmf-hier` generates the identical results with DC-NMF (but the former is less efficient) and is not shown in Fig. 5.3.

We can see that DC-NMF gives better cluster and topic quality in many cases, and improves the performance of HierNMF2 in every case. One possible reason for the better performance of DC-NMF is that documents that appear to be outliers are removed when building the hierarchy in HierNMF2, and thus the topics at the leaf nodes are more meaningful and represent more salient topics than those generated by a flat topic modeling method that takes every document into account. The algorithms solving NMF with separability constraints yielded the lowest clustering quality. Among them, `AnchorRecovery` and `Hottopixx` both require several parameters provided by the user, which could be time-consuming to tune and have a large impact on the performance of their algorithms. We used the default parameters for both of these methods, which may have negatively affected their NMIs.

5.5.2. Timing Results. Fig. 5.5 shows the run-time of the proposed methods versus NMF and K-means, all implemented in Matlab. DC-NMF required substantially less run-time compared to the standard flat NMF. These results show that flat clustering based on standard NMF exhibits a superlinear trend while hierarchical clustering based on Rank-2 NMF exhibits a linear trend of runtime as k increases. For example, to generate 70 clusters on the **Cora** data set, HierNMF2, DC-NMF, `nmf-hier`, and `nmf-flat` took about 2.4, 2.6, 5.6, and 55.3 minutes, respectively. We

[10] `http://math.ucla.edu/~dakuang/software/kmeans3.html`

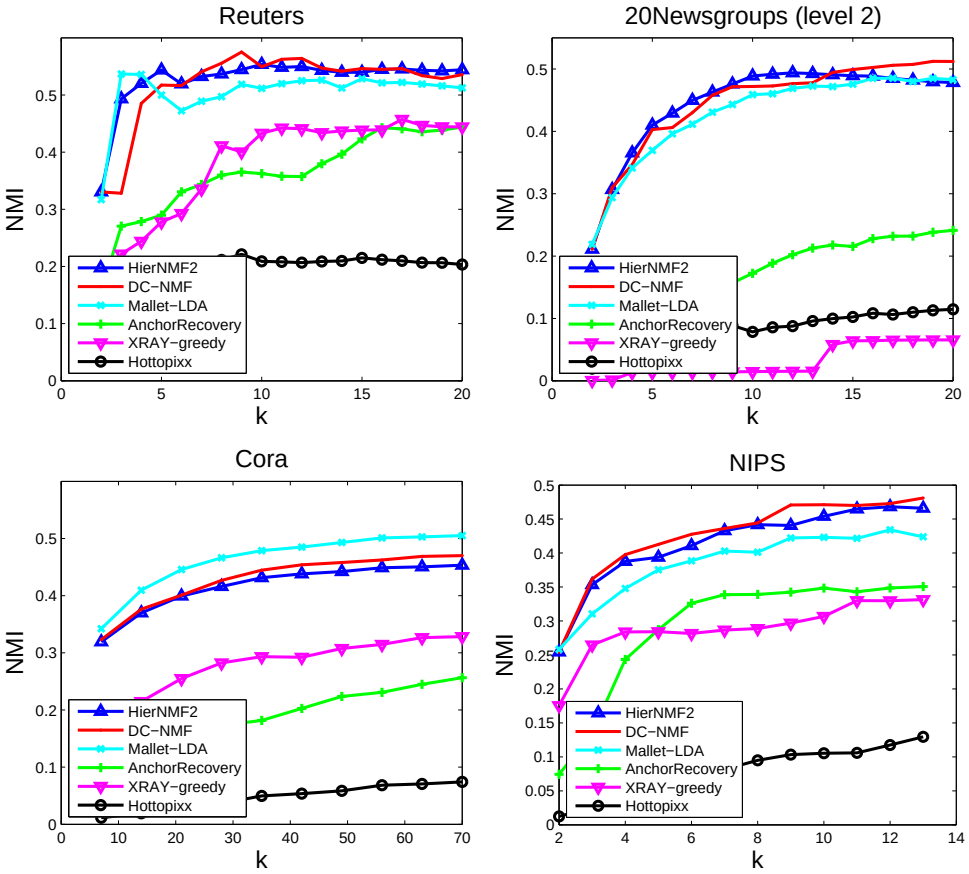

FIG. 5.4. *DC-NMF versus other* topic modeling methods *in cluster quality evaluated by normalized mutual information (NMI).*

note that K-means with only the batch-update phase has similar runtime to DC-NMF; however, the cluster quality is not as good, which was shown earlier in Fig. 5.3.

Fig. 5.6 compares the run-time of our C++ implementation of DC-NMF available in the software `smallk` [7] versus off-the-shelf toolkits (`CLUTO`, `Mallet-LDA`) and recent methods proposed for large-scale topic modeling, namely `AnchorRecovery`, `XRAY`, and `Hottopixx`. We used 8 threads when possible to set the number of threads manually (in the cases of `smallk`, `CLUTO`, `Mallet-LDA`, and `Hottopixx`).

On the **RCV1** and **Wiki-4.5M** data sets, DC-NMF is about 20 times faster than `Mallet-LDA`; particularly on the largest **Wiki-4.5M** data set in our experiments, DC-NMF found 80 topics in about 50 minutes, greatly enhancing the practicality of topic modeling algorithms when compared to the other software packages in our experiments.

The three algorithms `AnchorRecovery`, `XRAY`, and `Hottopixx` that solve NMF with separability constraints require a large $m \times m$ matrix, i.e. word-word similarities. We reduced the vocabulary of **Wiki-4.5M** to about 100,000 unique terms in order to accommodate the $m \times m$ matrix in main memory for these algorithms. Among them, `XRAY` and `Hottopixx` build a dense word-word similarity matrix and thus have a large memory footprint [33, 4]. `AnchorRecovery`, on the other hand, computes a random projection of the word-word similarity matrix, greatly reducing the time and space

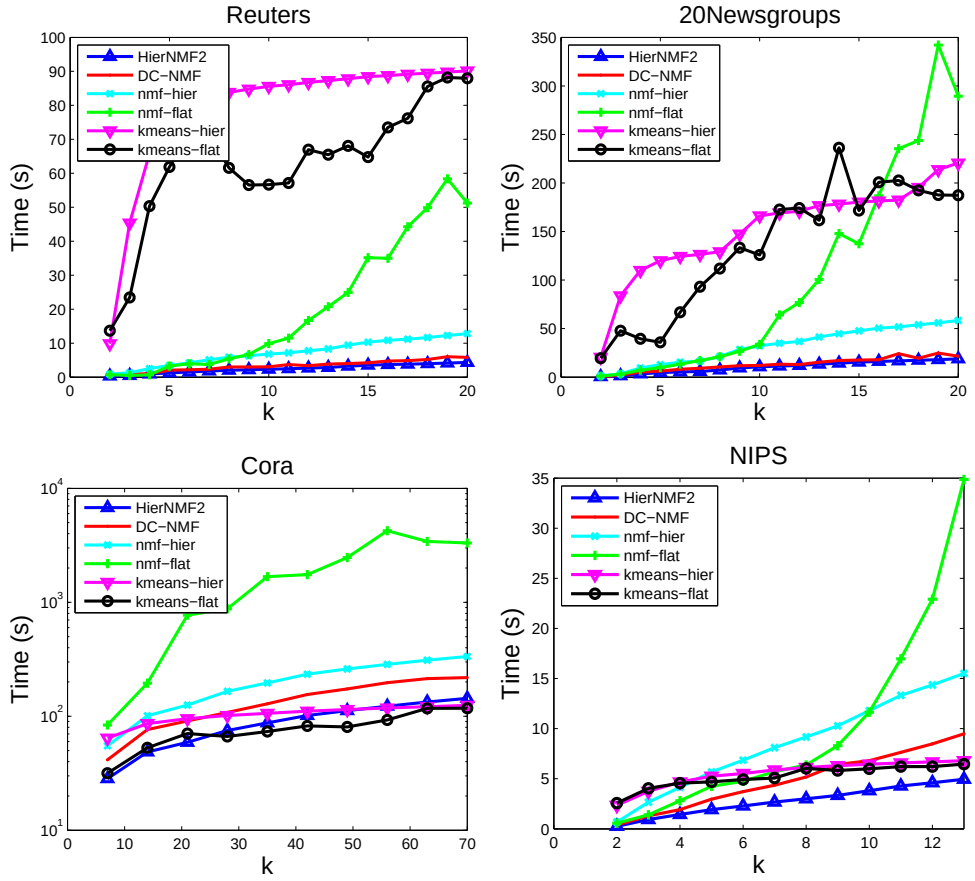

FIG. 5.5. *Timing results for the Matlab implementation of HierNMF2, DC-NMF, NMF, and K-means on the smaller data sets.*

complexity [1]; however, as we have seen in Fig. 5.4, its cluster quality is not as good as that of DC-NMF.

Overall, DC-NMF is the best-performing method in our experiments, considering both cluster quality and efficiency. The relatively recent software package `CLUTO` is also competitive.[11]

5.6. Illustration. To visualize the cluster/topic tree generated by HierNMF2, we show an illustration of the topic structure for a news article data set containing 100,361 articles in Fig. 5.7. First, we notice that the tree was not restrained to have a balanced structure, and HierNMF2 was able to determine the semantic organization on-the-fly. We can see that the articles were first divided into two big categories—politics/economy and art/entertainment/life. In the next few hierarchical levels, those topics (politics, economy, art, etc.) were further refined and emerged as more coherent sub-topics. Finally, at the leaf level, HierNMF2 produced fine-grained topics such as Iraq war, law and justice, stock market, movies, musics, health, houses and hotels.

[11]The run-time for `CLUTO` on **Wiki-4.5M** is absent: on our smaller system with 24 GB memory, it ran out of memory; and on our larger server with sufficient memory, the binary could not open a large data file (> 6 GB). The `CLUTO` software is not open-source and thus we only have access to the binary and are not able to build the program on our server.

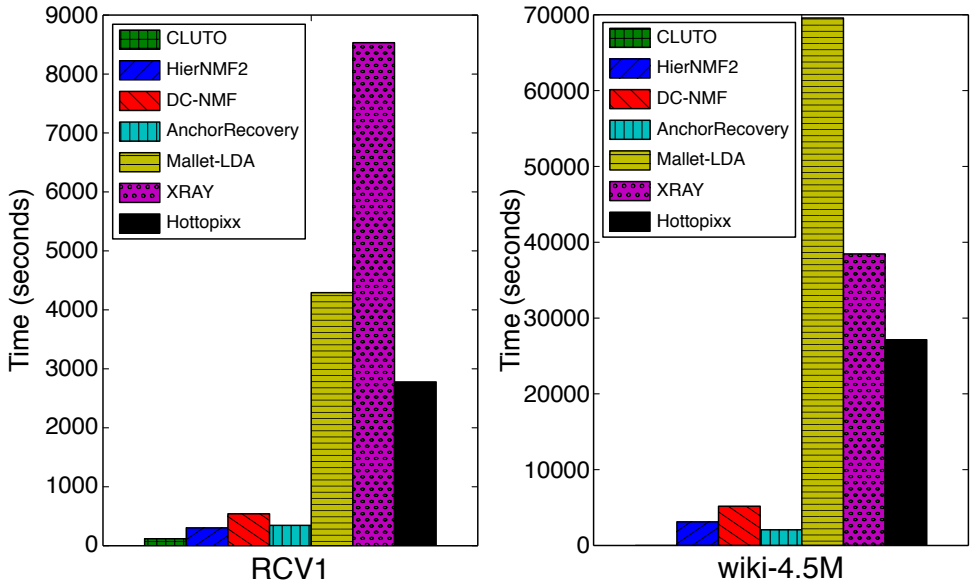

FIG. 5.6. *Timing results for the C++ implementation of HierNMF2 and DC-NMF available in our open-source software* `smallk` *and other state-of-the-art clustering and topic modeling methods on large, unlabeled text data sets.*

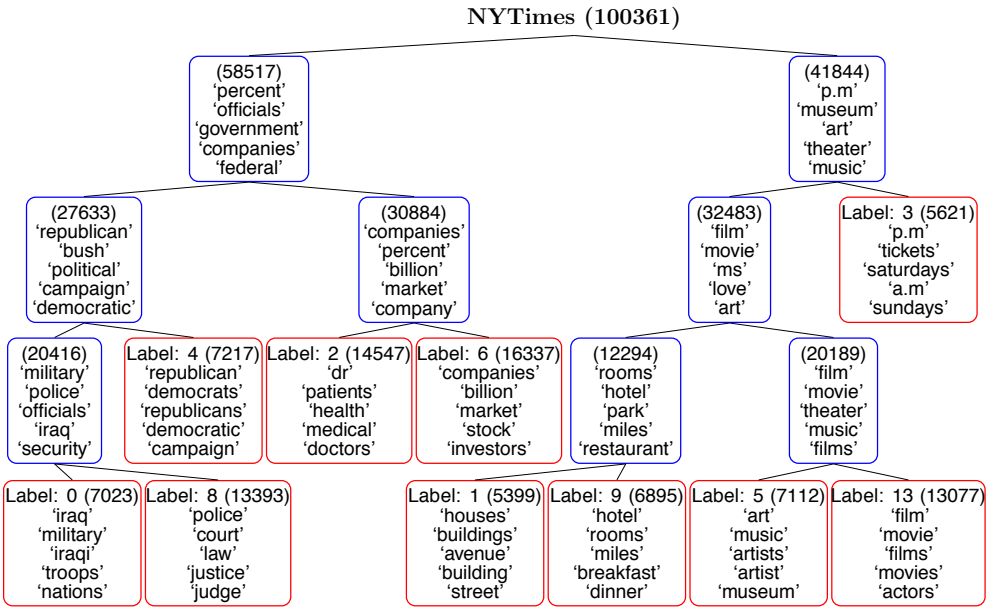

FIG. 5.7. *Hierarchical clustering result generated on a data set consisting of 100,361 New York Times articles for illustration. The hierarchy is automatically detected and not necessarily a balanced tree. Each tree node $\mathcal{N}$ is associated with a column of W, denoted as $\mathbf{w}_{\mathcal{N}}$, generated by Rank-2 NMF applied on its parent node. We display the five terms with highest importance values in $\mathbf{w}_{\mathcal{N}}$. Red boxes indicate leaf nodes while blue boxes indicate non-leaf nodes. The number in the parentheses at each node indicates the number of documents associated with that node.*

6. Conclusion. Clustering and topic modeling have become increasingly important tasks for big data analysis due to the explosion of text data and the need for extracting latent information from text corpora. Developing scalable methods for

modeling large-scale text resources efficiently has become necessary for studying social and economic behaviors, significant public health and safety issues, and network security, to name a few. Timely decisions based on actionable information derived from text sources is becoming much more critical in numerous domains.

In this paper, we proposed DC-NMF as a general NMF algorithm, with applications to large-scale clustering and topic modeling. The proposed approach is based on a divide-and-conquer strategy, exploiting the recent HierNMF2 method for constructing a binary tree using an efficient rank-2 NMF algorithm, and later flattening the tree structure into a flat partitioning of data points. We investigated various decision rules for choosing a leaf node to split in the hierarchy of topics, each with its advantage in computational efficiency or reconstruction quality. The results from HierNMF2 can be potentially applied to initializing other machine learning methods with iterative algorithms, such as K-means and total variational methods. We demonstrated the efficiency and effectiveness of DC-NMF as an algorithm for general rank-k NMF, and also as a text clustering and topic modeling method on data sets with ground-truth labels and larger unlabeled data sets. In our extensive tests, DC-NMF is over 100 times faster than standard NMF and about 20 times faster than LDA, and thus will have dramatic impacts on many fields requiring large-scale text analytics.

Acknowledgments. We would like to thank Dr. Yunlong He for Theorem 4.1. The work of the authors was supported in part by the National Science Foundation (NSF) grant IIS-1348152 and the Defense Advanced Research Projects Agency (DARPA) XDATA program grant FA8750-12-2-0309. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or the DARPA.

REFERENCES

[1] Sanjeev Arora, Rong Ge, Yoni Halpern, David M. Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu, *A practical algorithm for topic modeling with provable guarantees*, in ICML '13: Proc. of the 30th Int. Conf. on Machine Learning, 2013.

[2] Sanjeev Arora, Rong Ge, Ravindran Kannan, and Ankur Moitra, *Computing a nonnegative matrix factorization – provably*, in STOC '12: Proc. of the 44th Symp. on Theory of Computing, 2012, pp. 145–162.

[3] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, Belmont, MA, 1999.

[4] Victor Bittorf, Ben Recht, Christopher Re, and Joel Tropp, *Factoring nonnegative matrices with linear programs*, in Advances in Neural Information Processing Systems 25, NIPS '12, 2012, pp. 1214–1222.

[5] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum, *Hierarchical topic models and the nested Chinese restaurant process*, in Advances in Neural Information Processing Systems 16, 2003.

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan, *Latent Dirichlet allocation*, J. Mach. Learn. Res., 3 (2003), pp. 993–1022.

[7] Richard Boyd, Barry Drake, Da Kuang, and Haesun Park. `http://smallk.github.io/`, June 2016.

[8] Deng Cai, Xiaofei He, Jiawei Han, and T. S. Huang, *Graph regularized nonnegative matrix factorization for data representation*, IEEE Trans. on Pattern Analysis and Machine Intelligence, 33 (2011), pp. 1548–1560.

[9] Moody T. Chu and Matthew M. Lin, *Low-dimensional polytope approximation and its applications to nonnegative matrix factorization*, SIAM Journal on Scientific Computing, 30 (2008), pp. 1131–1155.

[10] Andrzej Cichocki and Anh Huy Phan, *Fast local algorithms for large scale nonnegative matrix and tensor factorizations*, IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences, E92A (2009), pp. 708–721.

[11] Joel E. Cohen and Uriel G. Rothblum, *Nonnegative ranks, decompositions, and factorizations of nonnegative matrices*, Linear Algebra and its Applications, 190 (1993), pp. 149 – 168.

[12] T.M. Cover and J.A. Thomas, *Elements of Information Theory*, Wiley-Interscience, Hoboken, NJ, 2nd ed., 2006.

[13] Barry Drake, Jingu Kim, Mahendra Mallick, and Haesun Park, *Supervised Raman spectra estimation based on nonnegative rank deficient least squares*, in Proceedings 13th International Conference on Information Fusion, Edinburgh, UK, 2010.

[14] Nicolas Gillis, *The why and how of nonnegative matrix factorization*, in Regularization, Optimization, Kernels, and Support Vector Machines, J.A.K. Suykens, M. Signoretto, and A. Argyriou, eds., Chapman & Hall/CRC, 2014, ch. 12, pp. 257–291.

[15] N. Gillis, Da Kuang, and Haesun Park, *Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization*, IEEE Transactions on Geoscience and Remote Sensing, 53 (2015), pp. 2066–2078.

[16] Amir Globerson, Gal Chechik, Fernando Pereira, and Naftali Tishby, *Euclidean embedding of co-occurrence data*, J. Mach. Learn. Res., 8 (2007), pp. 2265–2295.

[17] Gene H. Golub and Charles F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, 4th ed., 2013.

[18] L. Grippo and M. Sciandrone, *On the convergence of the block nonlinear Gauss-Seidel method under convex constraints*, Operations Research Letters, 26 (2000), pp. 127–136.

[19] Ngoc-Diep Ho, *Non-negative matrix factorization. Algorithms and applications*, PhD thesis, Universit catholique de Louvain, 2008.

[20] Thomas Hofmann, *Probabilistic latent semantic indexing*, in SIGIR '99: Proc. of the 22th Int. ACM Conf. on Research and development in informaion retrieval, 1999.

[21] Matan Hofree, John P Shen, Hannah Carter, Andrew Gross, and Trey Ideker, *Network-based stratification of tumor mutations*, Nature Methods, 10 (2013), pp. 1108–1115.

[22] Roger A. Horn and Charles R. Johnson, eds., *Matrix Analysis*, Cambridge University Press, New York, NY, USA, 1986.

[23] Cho-Jui Hsieh and Inderjit S. Dhillon, *Fast coordinate descent methods with variable selection for non-negative matrix factorization*, in 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '11), 2011, pp. 1064–1072.

[24] Anil K. Jain, *Data clustering: 50 years beyond k-means*, Pattern Recognition Letters, 31 (2010), pp. 651 – 666. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR).

[25] Hyunsoo Kim and Haesun Park, *Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis*, Bioinformatics, 23 (2007), pp. 1495–1502.

[26] ———, *Nonnegative matrix factorization based on alternating non-negativity-constrained least squares and the active set method*, SIAM J. on Matrix Analysis and Applications, 30 (2008), pp. 713–730.

[27] Jingu Kim, Yunlong He, and Haesun Park, *Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework*, Journal of Global Optimization, 58 (2014), pp. 285–319.

[28] Jingu Kim and Haesun Park, *Sparse nonnegative matrix factorization for clustering*, tech. report, Georgia Institute of Technology, 2008.

[29] Jingu Kim and Haesun Park, *Toward faster nonnegative matrix factorization: A new algorithm and comparisons*, in ICDM '08: Proc. of the 8th IEEE Int. Conf. on Data Mining, 2008, pp. 353–362.

[30] Jingu Kim and Haesun Park, *Fast nonnegative matrix factorization: An active-set-like method and comparisons*, SIAM J. on Scientific Computing, 33 (2011), pp. 3261–3281.

[31] D. Kuang and H. Park, *Fast rank-2 nonnegative matrix factorization for hierarchical document clustering*, in 19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '13), 2013, pp. 739–747.

[32] Da Kuang, Sangwoon Yun, and Haesun Park, *SymNMF: Nonnegative low-rank approximation of a similarity matrix for graph clustering*, J. Glob. Optim., (2014).

[33] Abhishek Kumar, Vikas Sindhwani, and Prabhanjan Kambadur, *Fast conical hull algorithms for near-separable non-negative matrix factorization*, in ICML '13: Proc. of the 30th Int. Conf. on Machine Learning, 2013.

[34] Daniel D. Lee and H. Sebastian Seung, *Learning the parts of objects by non-negative matrix factorization*, Nature, 401 (1999), pp. 788–791.

[35] ———, *Algorithms for non-negative matrix factorization*, in Advances in Neural Information

Processing Systems 14, NIPS '01, 2001, pp. 556–562.

[36] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li, *Rcv1: A new benchmark collection for text categorization research*, J. Mach. Learn. Res., 5 (2004), pp. 361–397.

[37] Liangda Li, Guy Lebanon, and Haesun Park, *Fast bregman divergence nmf using taylor expansion and coordinate descent*, in Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, New York, NY, USA, 2012, ACM, pp. 307–315.

[38] Chih-Jen Lin, *On the convergence of multiplicative update algorithms for nonnegative matrix factorization*, IEEE Trans. on Neural Networks, 18 (2007), pp. 1589–1596.

[39] Chih-Jen Lin, *Projected gradient methods for nonnegative matrix factorization*, Neural Computation, 19 (2007), pp. 2756–2779.

[40] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze, *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, 2008.

[41] Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore, *Automating the construction of Internet portals with machine learning*, Inf. Retr., 3 (2000), pp. 127–163.

[42] François Glineur Nicolas Gillis, *Accelerated multiplicative updates and hierarchical als algorithms for nonnegative matrix factorization*, Neural Computation, 24 (2012), pp. 1085–1105.

[43] A. Ozerov and C. Fvotte, *Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation*, Audio, Speech, and Language Processing, IEEE Transactions on, 18 (2010), pp. 550–563.

[44] P. Paatero and U. Tapper, *Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values*, Environmetrics, 5 (1994), pp. 111–126.

[45] M. H. Van Benthem and M. R. Keenan, *Fast algorithm for the solution of large-scale non-negativity constrained least squares problems*, J. Chemometrics, 18 (2004), pp. 441–450.

[46] Wei Xu, Xin Liu, and Yihong Gong, *Document clustering based on non-negative matrix factorization*, in SIGIR '03: Proc. of the 26th Int. ACM Conf. on Research and development in informaion retrieval, 2003, pp. 267–273.