Data Min Knowl Disc
DOI 10.1007/s10618-014-0347-0

Exemplar-based low-rank matrix decomposition for data clustering

Lijun Wang · Ming Dong

Received: 26 September 2011 / Accepted: 15 January 2014

Abstract Today, digital data is accumulated at a faster than ever speed in science, engineering, biomedicine, and real-world sensing. The ubiquitous phenomenon of massive data and sparse information imposes considerable challenges in data mining research. In this paper, we propose a theoretical framework, Exemplar-based low-rank sparse matrix decomposition (EMD), to cluster large-scale datasets. Capitalizing on recent advances in matrix approximation and decomposition, EMD can partition datasets with large dimensions and scalable sizes efficiently. Specifically, given a data matrix, EMD first computes a representative data subspace and a near-optimal low-rank approximation. Then, the cluster centroids and indicators are obtained through matrix decomposition, in which we require that the cluster centroids lie within the representative data subspace. By selecting the representative exemplars, we obtain a compact "sketch"of the data. This makes the clustering highly efficient and robust to noise. In addition, the clustering results are sparse and easy for interpretation. From a theoretical perspective, we prove the correctness and convergence of the EMD algorithm, and provide detailed analysis on its efficiency, including running time and spatial requirements. Through extensive experiments performed on both synthetic and real datasets, we demonstrate the performance of EMD for clustering large-scale data.

Responsible editor: Sugato Basu.

L. Wang (✉)
Department of Computer Science, Wayne State University,
5057 Woodward Avenue, Suite 3212, Detroit, MI 48201, USA
e-mail: ljwang@wayne.edu

M. Dong
Department of Computer Science, Wayne State University,
5057 Woodward Avenue, Suite 14110.1, Detroit, MI 48201, USA
e-mail: mdong@wayne.edu

Published online: 29 January 2014

Keywords Clustering · Low-rank approximation · Subspaces · Matrix decomposition

1 Introduction

With the rapid development of information technology, data storage has been made relatively inexpensive and bandwidth abundant, resulting in extremely large datasets from WWW, science and engineering simulations, and system sensors. Data mining provides us an effective way for the exploration and analysis of hidden patterns from these data for a broad spectrum of applications. Usually, these datasets share one prominent characteristic: tremendous in size with tens of thousands of objects and features. In addition, knowledge is very sparsely encoded because the patterns are usually active only in a local area. The ubiquitous phenomenon of massive data and sparse information imposes considerable challenges in data mining research. Recently, techniques that can expand the human ability to comprehend large-scale data have attracted significant attention in the research community.

Clustering is one of the most widely used data mining techniques, which refers to a variety of procedures designed to find natural groupings, or clusters, in multi-dimensional data, based on measured or perceived similarities among the patterns Jain et al. (1999) and Duda et al. (2001). The purpose of clustering is to extract useful information from unlabeled data. Among various clustering algorithms, k-means MacQueen (1967) is typically viewed as one of the simplest. However, for applications with high-dimensional data such as text mining Yen and Wu (2008) and Mahdavi and Abolhassani (2009), it can hardly achieve satisfactory clustering results. In these cases, dimensionality reduction techniques, which project the original data into a lower dimensional coordinate system, is widely employed to improve clustering efficiency and accuracy. For example, principle component analysis (PCA) Jolliffe (2002) can find a low-dimensional linear embedding of the data points that best preserves their variance as measured in the high-dimensional input space. Also working in a transformed space, kernel-based clustering has recently been proposed to discover nonlinear structures in the original data space Dhillon et al. (2005). Another popular clustering method is based on spectral graph partitioning, in which the data objects are modeled as vertices of a weighted graph with edge weights representing the similarities between two data objects. Clusters are then obtained by "cutting" the graph vertices into different partitions. As shown in Dhillon et al. (2005, 2004), the weighted kernel k-means can be solved by the normalized cut approach. Thus, both methods can obtain the partition of data by solving an eigenvalue problem where the clusters are inferred from the top eigenvectors.

More recently, matrix decomposition-based clustering has been proposed in the literature as an effective approach for clustering in high dimensional spaces Ding et al. (2008), Li and Ding (2006) and Drineas et al. (2004). Factor analysis gained by matrix decomposition provides another option to accomplish the goal of reducing the dimensionality and detecting the hidden structures of the data Berry et al. (2007). Nonnegative matrix factorization (NMF) based clusterings Berry et al. (2007), Lee and Seung (2000) and Wang et al. (2011) have been popularly applied in various

fields, such as text-mining Xu et al. (2003), image analysis Lee and Seung (1999) and bioinformatics Kim and Park (2007). By restricting the factors to be non-negative, NMF can generate interpretable low-rank factors: one represents the basis of the original data, and the other indicates the cluster probabilities. The interpretability of matrix factors is implicated by the relationship between NMF and k-means by making use of the least squares objectives Ding et al. (2008).

In order to analyze the data as a whole, all the aforementioned algorithms typically need much larger working space than the volume of the data. Though it is possible to solve this problem by using a online procedure or a parallel implementation, reliable decisions can only be made based on information sparsely encoded in the entire dataset Jain et al. (1999). Low-rank matrix approximation provides a powerful tool to analyze large-scale data. Since it provides a "sketch" of the original data Sun et al. (2008), operations on it lead to great computational and storage efficiency. In addition, low-rank approximations have the ability of extracting correlations and removing noise from matrix-structured data Achlioptas and Mcsherry (2007). Thus, incorporating low-rank approximation into clustering may lead to the improvement of clustering efficiency and accuracy.

The traditional hierarchical approaches and mixture-resolving clustering, like EM algorithm Dempster et al. (1977) and k-means MacQueen (1967), can hardly be integrated with low-rank approximations due to use of non-matrix structured data. Thus, low-rank approximations are more suitable in assisting matrix-based methods. For example, Nyström-based approximations have been widely applied in spectral clustering Fowlkes et al. (2004), Zhang and Kwok (2006) and Zhang et al. (2008), and kernel learning Drineas and Mahoney (2005) and Williams and Seeger (2001). Also, near-optimal low-rank approximation techniques are popularly used to speed up singular value decomposition (SVD) or eigenvalue decomposition in the projection-based clustering Berry et al. (2005) and Drineas et al. (2004). However, these approaches are performed in either a kernel space or transformed space. Thus, the results are not interpretable in the sense of cluster centroids. In addition, they do not take advantage of other desirable properties of low-rank approximations, such as the compact formation and noise reduction, to further improve the clustering performance.

In this paper, we propose Exemplar-based low-rank sparse matrix decomposition (EMD), a novel method for fast clustering large-scale data by incorporating low-rank approximations into matrix decomposition-based clustering. Following the previous work on the visualization of large document corpus Chen et al. (2009) and the preliminary version of this work in conference proceedings Wang and Dong (2011), we further extend our method on clustering large scale and sparse data. Specifically, given a data matrix A, EMD first computes a low-rank approximation $\tilde{A}$ and a representative data subspace C by applying near-optimal low-rank approximation methods. Next, the cluster centroids and indicators are obtained through the decomposition: $\tilde{A} = CWG^T$, where W is the weight matrix, and G is the cluster indicator matrix. In EMD, the cluster centroid matrix F equals CW, and each of its vectors is a linear combination of exemplars; for G, each entry indicates the posterior probability of a data point to a certain cluster.

Different from the methods mentioned above, EMD capitalizes on the highly attractive properties of low-rank approximation such as feature selection, robustness to noise,

and as a result, it can achieve a higher clustering accuracy. Moreover, if data can be faithfully represented in a low linear subspace, by constraining the cluster centroids within the representative data subspace selected by the low-rank approximation algorithm, EMD provides better interpretable clustering results than in the whole space. Finally, EMD gains sparse factors: the sparsity of the weight matrix means each cluster centroid is constructed based on the exemplars that are highly related to that cluster, and this also greatly strengthens the interpretability of clustering results; while a sparse indicator matrix means a sharp partition of the data. From a theoretical perspective, we prove the correctness and convergence of the EMD algorithm, and provide detailed analysis on its efficiency, including running time and spatial requirements. Through extensive experiments performed on both synthetic and real datasets, we demonstrate the performance of EMD for clustering high-dimensional and large data. The remainder of the paper is organized as follows. We first review related work in Sect. 2. Then, we present our algorithm and theoretical analysis in Sect. 3. In Section 4, we provide thorough experimental evaluation of EMD. Finally, we conclude in Sect. 5.

2 Related work

In this section, we introduce some essential background on clustering and low-rank matrix approximation and review related work in the literature.

2.1 Clustering

Clustering of real data, like online documents or gene expression data, has attracted great attention due to its potential applications in science, engineering, business, and biomedicine domains. Among a variety of clustering techniques Jain et al. (1999), we mainly overview the categories of mixture-based clustering, spectral clustering and matrix decomposition-based clustering.

Mixture-based clustering methods assume that the data items in a cluster are drawn from one of several distributions (usually Gaussian) and attempt to estimate the parameters of all these distributions. The introduction of the expectation maximization (EM) algorithm in Dempster et al. (1977) was an important step in solving the parameter estimation problem. k-means MacQueen (1967) is known as the most popular mixture density-based clustering algorithm with simplified computation and accelerated convergence. It performs iterative relocation to partition a dataset into k clusters, locally minimizing the overall distortion measurement between the data points and the cluster means (a.k.a., centroids). Note that, finding the global optima for the k-means objective function is an NP-complete problem Garey and Johnson (1979). Hence, in general, the k-means procedure will not converge to an optimal partition with a random initialization.

Clustering based on spectral graph partitioning Chung (1997) has been widely used in applications of machine learning and computer vision Hagen and Kahng (1992), Shi and Malik (2000) and Ding et al. (2001). Compared with k-means, spectral clustering is a matrix-based model, which is amenable to vigorous analysis and brings benefit from the well-established knowledge in linear algebra. It provides a relaxation version

of the partition problem and infers the clusters from the top eigenvectors of a graph's Laplacian matrix Fiedler (1973), derived from the similarity between data points. The two most common algorithms are Normalized Cuts Shi and Malik (2000) and Ratio Cut Hagen and Kahng (1992).

Matrix decomposition-based clustering has been shown to effectively cluster high dimensional data in text mining and multimedia data analysis. The SVD is one of most important matrix decompositions, with the general form of $X \approx U \Sigma V^T$, where U is a unitary basis consisting of left-singular vectors of X, V is a unitary basis consisting of right-singular vectors of X, and Σ is a diagonal matrix with singular-values on the diagonal. SVD decomposes a matrix with the constraint of orthogonality of U and V. In SVD, all the input data and factor matrices are allowed to have mixed-signs, so we can write,

$$SVD : X_{\pm} \approx U_{\pm} V_{\pm}, \tag{1}$$

when absorbing Σ into U. SVD-based clustering projects the input data into singular vector space, and then a traditional data clustering algorithm like k-means is applied to cluster the data in the transformed space. Since U and V can have negative entries, the translated data may have negative values though they are originally positive. Hence, for the positive input data like documents and images, the negativity of projected results lacks intuitive meaning.

As another popular matrix decomposition method, NMF has been given much attention recently and proved to be very useful in many applications. It is initially proposed for parts-of-whole decomposition Lee and Seung (1999), and later extended to a general framework for data clustering, in which nonnegative data matrices are decomposed into multiple nonnegative factors, containing the cluster posterior. Theoretically, NMF has been shown to be equivalent to the (kernel) k-means clustering Ding et al. (2005, 2008), probabilistic latent semantic indexing (PLSI) Ding et al. (2006) and the laplacian-based spectral clustering Ding et al. (2005). The standard NMF decomposes a nonnegative matrix X into two matrices F and G:

$$X \approx F G^T, \tag{2}$$

where $F \in R^{d \times k}$ and $G \in R^{k \times n}$ with the constraint that the two factors are nonnegative. From "parts-of-whole" view, F is known as the basis matrix Lee and Seung (1999, 2000), which contains a basis that is optimized for the linear approximation of the data in X. A good approximation can be achieved only if the basis vectors discover the latent structure in the data. G is an encoding matrix, and each column consists of the coefficients by which a data point is represented by a linear combination of the basis vectors. From clustering perspective, F can be regarded as the cluster centroid matrix with each column representing a cluster center, while G is the cluster indicator matrix with G_{ik} giving the posterior probability that $\mathbf{x}_i$ belongs to the k-th column cluster. The decomposition can be obtained by solving the following optimization problem:

$$J_{\text{NMF}} = \min_{F \geq 0, G \geq 0} \|X - F G^T\|^2, \tag{3}$$

where $\|.\|$ is the Frobenius norm.

Based on NMF, many variants have been developed in the literature for different purposes by adding more restrictions. For example, NMF with sparseness constraint is proposed to strengthen the parts-based representations Hoyer (2004); NMF restricted by orthogonality is to improve the interpretation of cluster indicators Ding et al. (2006); and kernel-NMF employs a kernel trick to improve the clustering accuracy Ding et al. (2008) and Xu and Gong (2004). However, in these methods, there is no constraint on the basis matrix F. That is, F can be anything in the solution space. As a result, the vectors of F are usually too far away from the true data distribution to be interpreted as cluster centroids. Moreover, these methods can only handle a nonnegative matrix.

To strengthen the connections between NMF and k-means, Ding et al. (2008) propose an interpretable matrix factorization method, Convex-NMF, by taking a constraint on the basis matrix F. Specifically, if the vectors of F are restricted to lie within the space spanned by the columns of X, the matrix factorization can be interpreted in terms of weighted cluster centroids. That is, if F is denoted as:

$$F = (\mathbf{f}_l, ..., \mathbf{f}_k), \tag{4}$$

then the restriction is:

$$\mathbf{f}_l = w_{1l}\mathbf{x}_1 + ... + w_{nl}\mathbf{x}_n = X\mathbf{w}_l. \tag{5}$$

This constraint has the advantage in the interpretation of the cluster centroid vector $\mathbf{f}_l$ as weighted sums (convex combination) of entire data points, and thus this restricted form of factorization is called Convex-NMF. The objective function of Convex-NMF is:

$$J_{\text{Convex}-\text{NMF}} = \min_{W\geq 0, G\geq 0} \|X - XWG^T\|^2, \tag{6}$$

where W is the weight matrix, and G is the indicator matrix. Compared with basic NMF, Convex-NMF establishes the relationship between the centroid space and the data space with $F = XW$, so it can generally capture the notion of cluster centroids. With this constraint, F can have negative elements if X is a mixed-sign matrix. Hence, Convex-NMF can be applied to both nonnegative and mixed-sign data matrices.

2.2 Low-rank matrix approximation

Low-rank matrix approximations have recently gained popularity in computer vision, information retrieval and machine learning. A widely used low-rank approximation is SVD, which is known to be optimal in the sense of achieving the minimum reconstruction error with Frobenius norm. However, the algorithms for computing the SVD generally operate through repeated matrix-vector multiplication, thereby requiring superlinear time and large working sets Golub and Loan (1996). Moreover, the result of SVD is usually dense even if the original matrix is sparse, so it is prohibitive to be applied on large-scale datasets Drineas et al. (2006). On the other hand, near-optimal low-rank approximations Achlioptas and Mcsherry (2007), a.k.a., exemplar-based approaches Tong et al. (2008), can preserve the sparsity of the matrix while achieving great savings on running time and space. For example, Algorithm 844 Berry et al. (2005)

applies Gram-Schmidt method in the pivoted QR decomposition to compute the *sparse column-row approximation*. In general, a pivoted QR factorization (PQR) has the form

$$AP = QT, \tag{7}$$

where A is the input matrix, P is the permutation matrix, Q is orthogonal and T is upper triangular. Further, Eq. (7) can be decomposed as

$$\left(B_1^{(k)} B_2^{(k)}\right) = \left(Q_1^{(k)} Q_2^{(k)}\right) \begin{pmatrix} T_{11}^{(k)} & T_{12}^{(k)} \\ 0 & T_{22}^{(k)} \end{pmatrix}, \tag{8}$$

where k is the rank, $B = AP = (B_1^{(k)} B_2^{(k)})$, and $B_1^{(k)}$ contains k columns. To compute a pivoted Q-less PQR factorization in Eq. (8), the Quasi-Gram-Schmidt algorithm is applied by successively selecting columns with the largest norm and updating the QR factorization of B. Consequently, a sparse column approximation to A can be obtained with the form $(B_1^{(k)} {T_{11}^{(k)}}^{-1})(T_{11}^{(k)} T_{12}^{(k)})$. To compute a column-row low-rank approximation, the Quasi-Gram-Schmidt algorithm is first applied to the columns of A to get a representative column set C and an upper triangular matrix T, and then applied to the rows of A to get a representative row set R and the corresponding upper triangular matrix S. The middle matrix U is computed as $U = T^{-1}T^{-T}(C^T AR)S^{-1}S^{-T}$ to minimize $\|A - CUR\|^2$ Stewart (1999). Finally, the column-row low-rank approximation can be obtained as,

$$\tilde{A} = CUR = C\left(T^{-1}T^{-T}(C^T AR)S^{-1}S^{-T}\right)R. \tag{9}$$

CUR is another well-known exemplar-based approximation method Drineas et al. (2006). It first selects the representative column and row exemplars as the left and right matrices according to their probability distributions and then computes the middle matrix based on these two matrices. Even though the original data may be too large to be completely loaded in the memory, CUR allows one to keep a small randomly-chosen and rapidly-computable "sketch" in RAM with a few passes over the original data stored in the disk. This promising property makes it possible to analyze tremendous amounts of data by studying its "sketch".

More recently, it has been shown that CMD Sun et al. (2008) provides a decomposition technique equivalent to the CUR algorithm while requiring less time and space. Different from CUR, CMD can generate a unique column subspace by carefully removing duplicate columns. The idea is to scale each unique sample up based on the square root of the number of times it is in the initial subspace. As proved, the newly produced unique subspace has the same singular values and left singular vectors as the subspace produced by CUR. Hence, CMD can achieve the equal accuracy with less computational and spatial costs. *Colibri* Tong et al. (2008) is another variant of CUR, which is proposed to solve the problem of *redundant* or *overcomplete* basis. That is, the columns of the initial subspace may be linearly dependent or near-duplicate, which is usually seen in a tightly-connected community where all nodes have similar

neighbors. Consequently, with the redundant basis, the approximation is not efficient in terms of space. In *Colibri*, a column initial subspace C_0 is first constructed by using the biased sampling strategy Drineas et al. (2006), which selects samples according to the distribution probability computed by the Frobenius norms. Then, a unique and independent subspace C is formed with an iterative procedure. In each iteration, a column in C_0 is checked to see if it is linearly dependent on the current columns of C. If not, this column is appended to C and the core matrix U is updated; else, this sample is discarded. Finally, C is obtained by eliminating all the redundant columns from C_0. In the algorithm, R is defined as $C^T A$, so the final approximation is $\tilde{A} = CUR$ with the representative column subspace C. All these near-optimal low-rank approximations are formed by the actual columns and rows, thus they can preserve the sparsity of the original matrix and ensure the spatial savings.

2.3 Clustering with low-rank approximation

Low-rank approximations have been successfully applied in many clustering algorithms to improve the efficiency. One of the most popular applications is to approximate the SVD in the projection-based clustering, where the transformation procedure is usually time-consuming. In Berry et al. (2005), proposed a variant of the 844 algorithm for approximating the SVD. Specifically, when rewriting the form of $\tilde{A}$ in Eq. (9), we get

$$\tilde{A} = (CT^{-1})(T^{-T}C^T ARS^{-1})(S^{-T}R) = PW^T Q, \tag{10}$$

where $P = CT^{-1}$, $W = T^{-T}C^T ARS^{-1}$, and $Q = S^{-T}R$. If the SVD of W is $W = M\Sigma N^T$, then by setting $V_1 = PM$ and $V_2 = QN$, we can get the approximate SVD of A as,

$$\tilde{A} = V_1 \Sigma V_2^T. \tag{11}$$

Notice P and Q are orthogonal according to QR factorization, so are V_1 and V_2. Another well known SVD approximation algorithm is the randomized SVD in Drineas et al. (2004). Given a matrix A, the randomized SVD first constructs a submatrix C by picking a set of columns of A according to the column probability distribution, then after scaling C properly, computes its left singular vectors and singular values to approximate the SVD of the whole matrix.

Recently, the Nyström-based low-rank approximation has been popularly employed in assisting the clustering of large-scale data. The Nyström methods are mainly developed with the numerical integration theory Baker (1997) and Delves and Mohamed (1985), which uses the randomly selected samples to approximate the affinity matrix and the eigenvectors of the graph's Laplacian. It is shown Nyström-based spectral clustering is empirically efficient to segment images Fowlkes et al. (2004) and clustering large datasets Yan et al. (2009). Also, in Drineas and Mahoney (2005) and Williams and Seeger (2001), the Nyström-based low-rank approximation is used to approximate the gram matrix to improve the kernel learning.

Table 1 Symbol definitions

Symbol	Definition
$A_{d\times n}$	Data matrix with the size of $d \times n$
$\tilde{A}_{d\times n}$	The approximation of the matrix A with the size of $d \times n$
$A^T, C^T, R^T \ldots$	The transpose of a matrix
X^{-1}	The inverse of a matrix X
$A(i, j)$	The entry(i,j) of A
$A(i, :)$	The ith row of A
$A(:, j)$	The jth column of A
c	The number of the selected samples (the size of data subspace)
k_c	The number of the clusters
$C_{d\times c}$	The column representative matrix (the left matrix of the approximation)with the size of $d \times c$
$R_{c\times n}$	The right matrix of the approximation with the size of $c \times n$
$U_{c\times c}$	The middle matrix of the approximation
$W_{c\times k_c}$	The weight matrix with the size of $c \times k_c$
$G_{k_c\times n}$	The indicator matrix with the size of $k_c \times n$

3 EMD for clustering

In this section, we present the EMD algorithm in detail. We first present the model formulation and algorithm details in Sect. 3.1, then provide the theoretical results in Sect. 3.2, including the proofs of correctness and convergence, the analysis of time and space complexities, and the advantages of EMD when compared with other clustering methods.

3.1 Model formulation and algorithm

We formulate data clustering as a matrix decomposition problem (all symbols used are listed in Table 1). To solve this problem, we propose to take a two-step approach. First, a low-rank approximation method is employed to select the representative data subspace and generate the compact approximation to the original data. Specifically, given a data matrix $A_{d\times n}$, its low-rank approximation is generally denoted as

$$\tilde{A} = CUR. \tag{12}$$

If A is a feature-object matrix, ($A(i, j)$ denotes the ith feature of the jth object), then C is regarded as a representative data subspace, which contains a set of columns selected from A, known as data exemplars. Similarly, R can be seen as a representative feature subspace if constructed by selecting a set of rows from A. Alternatively, R can also be computed from C, e.g., $R = C^T A$ Tong et al. (2008). Finally, the middle matrix U is computed by minimizing $\|A - CUR\|_F^2$. Thus, at the end of the first step, we obtain the low-rank approximation $\tilde{A}$ and the representative data exemplar space C.

In the second step, we need to get the cluster centroids and cluster indicators in the low-rank exemplar space. We formulate this task as an optimization problem,

$$\begin{aligned} J &= \min_{W\geq 0, G\geq 0} \|\tilde{A} - CWG^T\|^2 \\ &= \min_{W\geq 0, G\geq 0} \mathrm{Tr}\left(\tilde{A}^T\tilde{A} - \tilde{A}^T CWG^T - GW^T C^T\tilde{A} + GW^T C^T CWG^T\right), \end{aligned} \tag{13}$$

where W is the weight matrix and G is the cluster indicator matrix. In the optimization, we propose an iterative algorithm to get nonnegative W and G while fixing arbitrarily signed C and $\tilde{A}$. The updating rules are obtained by using the auxiliary functions and the optimization theory Ding et al. (2008):

$$W_{(i,h)} \leftarrow W_{(i,h)} \sqrt{\frac{(P_1{}^+G)_{(i,h)} + (P_3{}^-WG^TG)_{(i,h)}}{(P_1{}^-G)_{(i,h)} + (P_3{}^+WG^TG)_{(i,h)}}}, \tag{14}$$

$$G_{(i,h)} \leftarrow G_{(i,h)} \sqrt{\frac{(P_2{}^+W)_{(i,h)} + (GW^TP_3{}^-W)_{(i,h)}}{(P_2{}^-W)_{(i,h)} + (GW^TP_3{}^+W)_{(i,h)}}}. \tag{15}$$

Details of EMD are provided in Algorithm 1. Note that in this formulation, the cluster centroid matrix F equals CW. That is, each cluster centroid is a linear combination of weighted exemplars, which makes the clustering results highly interpretable. Besides, we notice that the computations involved are mainly in the matrix multiplication step, and it is possible to perform the matrix multiplication in EMD in parallel.

When the data matrix is very large, one can only select a very small amount of samples limited by the memory. The sparsity of the matrix is another important factor that affects the sample selection. Clearly, the sparser the matrix is, the more samples can be selected. For example, for a dataset with four million data points, ten features and 0.01 % density, at most a few dozens of samples could be selected on a PC with 3GB RAM and the Windows XP operating system. This number is further reduced when the dataset has a large number of features.

In the data approximation, i.e., $A = CUR$, we notice R requires large space when the dataset contains large amount of data points. That is, when the number of samples increases, the spatial cost increases greatly. To solve this problem, when R is sparse, it may be approximated by approximate subspace methods. In EMD-C, R is computed by C^TA. So, it is dense, and approximate subspace will not help in this case. In EMD-QR, R is constructed by the real columns from the original sparse matrix. So, R is sparse, and it is possible to employ approximate subspace methods to improve spatial efficiency. In the literature, several approximate subspace methods have been discussed Shyamalkumar and Varadarajan (2007). In addition, all the low-rank matrix approximation algorithms mentioned in this paper can also provide an approximate subspace. When the approximation to R, i.e., $\tilde{R}$, is computed, it can be plugged in the updating rules (Eqs. 14, 15) accordingly.

Algorithm 1 EMD

INPUT: $A \in \mathbb{R}^{d\times n}$, $k_c \in \mathbb{Z}^+$ s.t. $1 \le k_c \le n$
OUTPUT: $W \in \mathbb{R}^{c\times k_c}$, $G \in \mathbb{R}^{n\times k_c}$

1. With input A, a near-optimal low-rank approximation method is used to get $\tilde{A} = CUR$ and the data exemplar subspace C;
2. Initializing W and G with non-negative random numbers, which are pseudorandom, scalar values drawn from a normal distribution with zero mean and unit standard deviation;
3. Iterate by using the following updating rules for each i and h until convergence;
   (a) Let $P_1 = C^T\tilde{A}$, $P_2 = \tilde{A}^T C$ and $P_3 = C^T C$, then split each matrix into the positive and negative parts:

$$P_i^+ = (|P_i| + P_i)/2; \qquad P_i^- = (|P_i| - P_i)/2;$$

   where $i \in \{1, 2, 3\}$;
   (b)

$$W_{(i,h)} \leftarrow W_{(i,h)} \sqrt{\frac{(P_1{}^+G)_{(i,h)} + (P_3{}^-WG^TG)_{(i,h)}}{(P_1{}^-G)_{(i,h)} + (P_3{}^+WG^TG)_{(i,h)}}},$$

$$G_{(i,h)} \leftarrow G_{(i,h)} \sqrt{\frac{(P_2{}^+W)_{(i,h)} + (GW^TP_3{}^-W)_{(i,h)}}{(P_2{}^-W)_{(i,h)} + (GW^TP_3{}^+W)_{(i,h)}}}.$$

3.2 Theoretical analysis

In this section, we first prove that our algorithm is correct and converges under the updating rules given in Eqs. (14), (15). In addition, we show the efficiency of EMD by analyzing the space and time required in computation. Finally, we point out the advantages of EMD when compared with other clustering methods.

3.2.1 Correctness and convergence of EMD

In Ding et al. (2008), it was shown that fixing G, under the update rules for W in Convex-NMF, the residual $\|X - XWG^T\|^2$ decreases monotonically, and the solution converges to a KKT fixed point. Here, we prove the correctness and convergence of EMD, following the idea used in the proof in Ding et al. (2008) but with different objectives and auxiliary functions.

To prove the correctness of EMD, we first introduce the Lagrangian function that satisfies the KKT complementary conditions. By setting its gradients to zeros, we obtain the fixed point equations, whose solutions must converge to a stationary point. If we can show that the updating rules in Eqs. (14), (15) satisfy these fixed point equations as well as the KKT fixed point condition, the correctness of EMD is proved. Specifically, the correctness of EMD can be stated as,

Proposition 1 *(Correctness of EMD) Given the objective function of Eq. (13), the constrained solution satisfies the K K T complementary conditions under the updating rules in Eqs. (14), (15).*

Proof To solve the optimization problem, we introduce the Lagrangian function

$$\begin{aligned} L(W, G, \lambda_1, \lambda_2) &= \|\widetilde{A} - CWG^T\|_F^2 - \mathrm{Tr}(\lambda_1 W^T) - \mathrm{Tr}(\lambda_2 G^T) \\ &= \mathrm{Tr}\left[(\tilde{A} - CWG^T)^T(\tilde{A} - CWG^T) - \lambda_1 W^T - \lambda_2 G^T\right] \\ &= \mathrm{Tr}\Big[(\tilde{A}^T\tilde{A} - \tilde{A}^T CWG^T - GW^T C^T\tilde{A} \\ &\quad + GW^T C^T CWG^T) - \lambda_1 W^T - \lambda_2 G^T\Big], \end{aligned} \tag{16}$$

where λ_1 and λ_2 are Lagrangian multipliers with nonnegative values, which constrain the nonnegativity of W and G, respectively. This function satisfies KKT complementary conditions. By setting the gradients to zeros, we obtain the following equations:

$$\frac{\partial L}{\partial W} = -2C^T\tilde{A}G + 2C^T CWG^T G - \lambda_1 I = \mathbf{0}, \tag{17}$$

$$\frac{\partial L}{\partial G} = -2\tilde{A}^T CW + 2GW^T C^T CW - \lambda_2 I = \mathbf{0}. \tag{18}$$

From the complementary conditions, we obtain:

$$\left(-2C^T\tilde{A}G + 2C^T CWG^T G\right)_{(i,h)} W_{(i,h)} = \lambda_1 W_{(i,h)} = 0, \tag{19}$$

$$\left(-2\tilde{A}^T CW + 2GW^T C^T CW\right)_{(i,h)} G_{(i,h)} = \lambda_2 G_{(i,h)} = 0. \tag{20}$$

These are fixed point equations, and the solutions must eventually converge to a stationary point. From the above two equations, we derive another two equal equations:

$$\left(-2C^T\tilde{A}G + 2C^T CWG^T G\right)_{(i,h)} W^2_{(i,h)} = 0, \tag{21}$$

$$\left(-2\tilde{A}^T CW + 2GW^T C^T CW\right)_{(i,h)} G^2_{(i,h)} = 0. \tag{22}$$

At convergence, $W^{t+1} = W^t$ and $G^{t+1} = G^t$, where t is the number of iterations. Hence, the constrained solution with updating rules in Eqs. (14) and (15) satisfies Eqs. (21) and (22) as well as the KKT fixed point condition. The proof is completed.

To prove the convergence of EMD, we need to construct two auxiliary functions with respect to W and G respectively, with which the object function of Eq. (13) is monotonically decreasing under the updating rules. Specifically, the convergence of EMD can be stated as,

Proposition 2 *(Convergence of EMD) The object function of Eq. (13) is monotonically decreasing under the updating rules in Eqs. (14), (15).*

Proof We construct auxiliary functions to prove that Eq. (13) decreases monotonically under the updating rules.

An auxiliary function $Z(X^{t+1}, X^t)$ should satisfy the two conditions:

$$Z(X^{t+1}, X^t) \geq J(X^{t+1}), \qquad Z(X^t, X^t) = J(X^t), \tag{23}$$

for any X^{t+1} and X^t. We define

$$X^{t+1} = \min_X Z(X, X^t). \tag{24}$$

Then we obtain the following equations:

$$J(X^t) = Z(X^t, X^t) \geq Z(X^{t+1}, X^t) \geq J(X^{t+1}). \tag{25}$$

Thus, with a proper auxiliary function, $J(X^t)$ is decreasing monotonically. Now, we construct the auxiliary functions with respect to W and G. In Eq. (13), $\tilde{A}^T\tilde{A}$ is a constant matrix, so in the following equations, we omit it. Let $X = W$, $B = C^T\tilde{A}G$, $P = C^TC$ and $Q = G^TG$, then the objective function with W is

$$J(X) = \mathrm{Tr}(-2X^TB + X^TPXQ). \tag{26}$$

Since B and P are mixed-sign matrices, we rewrite $J(X)$ by splitting each mixed-sign matrix into positive and negative parts:

$$J(X) = \mathrm{Tr}(-2X^TB^+ + 2X^TB^- + X^TP^+XQ - X^TP^-XQ). \tag{27}$$

Then, we construct an auxiliary function for $J(X)$ as:

$$\begin{aligned} Z(X', X) = &-2\sum_{ik} B^+_{ik}X'_{ik}\left(1 + log\frac{X_{ik}}{X'_{ik}}\right) + \sum_{ik} B^-_{ik}\frac{X^2_{ik} + X'^2_{ik}}{X'_{ik}} + \sum_{ik}\frac{(P^+X'Q)_{ik}X^2_{ik}}{X'_{ik}} \\ &- \sum_{ijkl} P^-_{ij}X'_{jk}Q_{kl}X'_{il}\left(1 + log\frac{X_{jk}X_{il}}{X'_{jk}X'_{il}}\right), \end{aligned} \tag{28}$$

where the subscription ik is the index of an element in the matrix. To get its global minimum, we take $\frac{\partial Z(X', X)}{\partial X_{ik}} = 0$ and get the updating rule for W as Eq. (14). Similarly, when $X = G$, $B = \tilde{A}^TCW$ and $P = W^TC^TCW$, the objective function with G is

$$J(X) = \mathrm{Tr}\left(-2X^TB^+ + 2X^TB^- + XP^+X^T - XP^-X^T\right). \tag{29}$$

So, its auxiliary function is

$$\begin{aligned} Z(X', X) = &-2\sum_{ik} B^+_{ik}X'_{ik}\left(1 + log\frac{X_{ik}}{X'_{ik}}\right) + \sum_{ik} B^-_{ik}\frac{X^2_{ik} + X'^2_{ik}}{X'_{ik}} + \sum_{ik}\frac{(P^+X')_{ik}X^2_{ik}}{X'_{ik}} \\ &- \sum_{ikl} P^-_{kl}X'_{ik}X'_{il}\left(1 + log\frac{X_{ik}X_{il}}{X'_{ik}X'_{il}}\right). \end{aligned} \tag{30}$$

By taking $\frac{\partial Z(X',X)}{\partial X_{ik}} = 0$, we get the updating rule for G as Eq. (15). Hence, with updating W and G according to Eqs. (14) and (15) in the EMD algorithm, the auxiliary functions will decrease monotonically until convergence, which also holds for W and G. The proof is completed.

3.2.2 Time and space complexity

To cluster a large dataset, efficiency in both space and speed is essential. In the following, we provide the analysis on the time and space complexity of EMD. First, in Algorithm 1, the computation of the low-rank matrix approximation is highly efficient. For example, the 844 algorithm is empirically shown faster than SVD for tens of times Berry et al. (2005). Further, in most low-rank approximation algorithms, the running time mainly depends on the number of selected samples c, e.g., the time complexity of *Colibri* is $\mathcal{O}(nc + c^3)$ Tong et al. (2008), and c is typically far less than n.

In the decomposition step, even though $\tilde{A}$ is used in the description of the algorithm, the computation is actually done using the three small matrices, C, U and R. This is also the base for our time analysis. For matrix decomposition, we first need to compute P_1, P_2 and P_3 with the following time:

$$\begin{aligned} P_1 &: \mathcal{O}(c(d \times c + c \times c + c \times n)), \\ P_2 &: \mathcal{O}(c(d \times c + c \times c + c \times n)), \\ P_3 &: \mathcal{O}(c^2 d), \end{aligned}$$

where c is the size of data subspace, and d and n are the size of the input matrix. Next, we need to compute W and G in Eqs. (14) and (15). With one iteration the time is

$$\begin{aligned} W &: \mathcal{O}(c^2 k_c + c k_c^2 + k_c^2 n + c n k_c), \\ G &: \mathcal{O}(c^2 k_c + c k_c^2 + k_c^2 n + c n k_c), \end{aligned}$$

where k_c is the number of clusters. Let t be the number of iterations, then P_1, P_2 and P_3 are computed only once, and W and G have to be calculated for t times, so the total time is $\mathcal{O}(3c^2 d + 2c^3 + 2c^2 n + t(cnk_c + c^2 k_c + ck_c^2 + k_c^2 n))$. Because $k_c \ll min(c, d, n)$ and $c \ll min(d, n)$, the overall time complexity is $\mathcal{O}(c^2 m + tcnk_c)$, where $m = max(d, n)$.

Regarding the space complexity, a low-rank approximation algorithm usually requests very small working space to fast extract a compact representation of the whole data. There are various methods proposed in the literature that can compute the approximation without loading all the data into the memory, such as CUR Drineas et al. (2006) and Nyström-based algorithms Drineas and Mahoney (2005) and Williams and Seeger (2001). Thus, the spatial cost of the first step in EMD depends on the choice and implementation of the approximation algorithm. In the decomposition step, EMD first needs $NNZ(C) + NNZ(U) + NNZ(R)$ units to store the approximation ($NNZ(.)$ represents the non-zero entries in a matrix). Since $NNZ(C) \leq cd$, $NNZ(R) \leq cn$, $NNZ(U) = c^2$, and $c \ll min(d, n)$, we have $\mathcal{O}(c(d + n))$. In addition, to solve the non-negative quadratic optimization problem, EMD needs ck_c and nk_c units for

storing W and G, respectively. The temporal storage for computing P_i and updating W and G requires $\mathcal{O}(cn)$ units. Generally, $k_c \ll min(c, d, n)$. Thus, the total space used is $\mathcal{O}(c(d + n))$.

Notice that the spatial requirement of storing the approximation and that of running the clustering algorithm are of the same order. That is, the overall space complexity of EMD is the same as the storage requirement for the approximation. Thus, we use the relative storage of the approximation results as a measure of the spatial cost of EMD, i.e., $SPCost = \frac{NNZ(\tilde{A})}{NNZ(A)}$. Also note that the size of data subspace c plays an important role in determining the computational and spatial costs. The lower c is, the less running time and space EMD uses.

3.2.3 Advantages of EMD

From a theoretical point of view, EMD uniquely combines matrix approximation and decomposition to provide a general framework for efficient data clustering. Under the EMD framework, various methods such as Algorithm 844 Berry et al. (2005), CUR Drineas et al. (2006), CMD Sun et al. (2008) and Colibri Tong et al. (2008) can be used to generate the approximation to the data matrix. In the decomposition step, EMD achieves the clustering by solving a non-negative quadratic optimization problem, which can be computed using an iterative approach. In EMD, we derive the factors based on the well-known KKT complementary conditions and auxiliary functions Lee and Seung (1999, 2000) and Xu et al. (2003). Note that other optimization algorithms, like gradient based methods and multiplicative updates Sha et al. (2007) and Xu and Gong (2004), can also be adopted. Further, existing approaches such as NMF and Convex-NMF can be considered as special cases of EMD. Specifically, if we skip the matrix approximation step and assume W is a unity matrix and C equals F, EMD becomes NMF, and EMD is same as Convex-NMF when it must use the entire data space for clustering. Finally, just like NMF and Convex-NMF, EMD performs a soft clustering and can be considered as a relaxation of k-means Ding et al. (2005).

Compared with NMF and Convex-NMF, EMD has the following unique features:

- Accuracy and Robustness: EMD capitalizes on the robustness from low-rank matrix approximation, which is very effective to remove Gaussian noise in a data set Achlioptas and Mcsherry (2007). In addition, nearly optimal low-rank approximation is employed in EMD. It selects the representative exemplars to form a "clean" sketch for the entire dataset and constrain the clustering centroids in the representative subspace. Thus, the negative effect of data correlation and noise is further reduced, leading to more accurate and robust clustering results.
- Interpretability: EMD selects a set of data exemplars as the representative subspace C. Compared with NMF in which no constraint is imposed on the centroid matrix and Convex-NMF that requires the cluster centroids to lie within the entire sample space, EMD guarantees the cluster centroids F to lie within C. This result nicely captures the notion of the cluster "centroids": the weighted sums of the representative samples.
- Sparsity: The near-optimal low-rank approximation can preserve the sparsity of data. Consequently, EMD can efficiently process extremely large sparse datasets.

In addition, similar with NMF and Convex-NMF, EMD generates sparse factors. The sparsity of weight matrix W leads to the construction of cluster centroids with highly related exemplars, while the sparsity of the cluster indicator matrix G provides a sharpe partition of the data.

– Efficiency: EMD provides an economic approach to compute the clustering of datasets. Compared with $\mathcal{O}(tdnk_c)$ time complexity in NMF and $\mathcal{O}(n^2d + tn^2k_c)$ in Convex-NMF, EMD is very efficient with $\mathcal{O}(c^2m + tcnk_c)$ $(m = max(d, n))$ since c is typically much smaller than d and n. Regarding the space complexity, EMD needs $\mathcal{O}(c(d + n))$ space, which may be less than the storage requirement for the original data.

4 Experimental results

In this section, we evaluate the performance of EMD on both synthetic and large-scale real datasets in terms of clustering quality, running time, and spatial costs by comparing it with leading clustering methods. All algorithms were implemented using MATLAB 7. The experiment was performed on a machine with a 3.0GHz Intel Xeon CPU, 3.0GB RAM and the Windows XP operating system. All the results reported are averaged over 10 runs.

4.1 Evaluation methods

In our experiments, we implemented two algorithms under the EMD framework by applying different low-rank approximation algorithms. The first one is EMD-C in which the Colibri method Tong et al. (2008) is used to select exemplar columns and provide a unique, linearly independent subspace. The second one, EMD-QR, employs the 844 algorithm Berry et al. (2005), which selects both columns and rows from the data matrix to produce a sparse column-row approximation. In order to evaluate our EMD approach, we compare both the algorithms with leading matrix-based clustering methods, i.e., NMF, Convex-NMF (CN), and tri-factorization (CoNMF) Ding et al. (2006). In CoNMF, the rows and columns of the data matrix are clustered into the same number of categories simultaneously, i.e., the number of clusters of data points. Besides, we also compare the performance of EMD with the baseline clustering algorithm: k-means (Kmeans) and the project-based clustering (SVDC), which first projects the original data to the space spanned by the top singular vectors, and then clusters the transformed data using k-means. In SVDC, the approximation method in Drineas et al. (2004) is adopted to obtain the top few singular vectors and the corresponding singular values to speed up SVD on large-scale datasets. In summary, overall we have seven clustering methods compared in our experiments, i.e., EMD-C, EMD-QR, CN, NMF, CoNMF, Kmeans and SVDC. For a fair comparison, we fix the number of iterations if iterative optimization is involved in an algorithm. As algorithms such as Convex-NMF and NMF run very slow on large-scale datasets, in our experiments, we followed the procedure used in Ding et al. (2008) (CN) and set the number of iterations to 100.

In the EMD algorithm, the clustering results depend on the choice of the subspace. For the sake of saving computational and spatial costs, we want to select the smallest c that keeps the matrix approximation error under a given tolerance level tol: $\|A-\tilde{A}\|^2 < tol$. In addition, we also specify a c_{max} which puts a upper bound on the total number of columns that can be selected. Specifically, in our experiments we have

$$tol = \alpha \|A\|^2, \tag{31}$$

where α is a value in [0,1].

All our comparisons are conducted using the following evaluation metrics:

1. Normalized mutual information (NMI) Strehl et al. (2002): The NMI value is computed from the confusion matrix based on the true and predicted cluster labels,

$$NMI = \frac{\sum_{h=1}^{k^{(a)}} \sum_{l=1}^{k^{(b)}} n_{h,l} log\left(\frac{n \cdot n_{h,l}}{n_h^{(a)} n_l^{(b)}}\right)}{\sqrt{\left(\sum_{h=1}^{k^{(a)}} n_h^{(a)} log\frac{n_h^{(a)}}{n}\right)\left(\sum_{l=1}^{k^{(b)}} n_l^{(b)} log\frac{n_l^{(b)}}{n}\right)}}, \tag{32}$$

   where $k^{(a)}$ and $k^{(b)}$ are cluster numbers in the true and predicted clustering results, $n_{h,l}$ the element in the confusion matrix, $n_h^{(a)}$ and $n_l^{(b)}$ the number of objects in the hth cluster of the true and predicted clustering, respectively. NMI ranges in [0, 1]. A high NMI value indicates that the predicted clustering matches the ground truth well.
2. Clustering accuracy: We evaluate the clustering accuracy Ac with the known class labels Ding et al. (2008): the confusion matrix is first computed; then the rows and columns are reordered so that the sum of the diagonal achieves the maximum; finally, we compute Ac as

$$Ac = \frac{\sum_{i=1}^{k_c} M_{conf}(i,i)}{n}, \tag{33}$$

   where M_{conf} is the confusion matrix, and n is the total number of data points.
3. Computational cost: We use the inline functions of MATLAB, tic and toc, to compute the running time. For EMD, we count the running time for both the approximation and decomposition procedures.
4. Spatial cost: We compute the storage of the approximation results as the spatial cost of EMD (normalized based on the required storage units for the original data matrix). In Algorithm 1, the approximation matrix is denoted as $\{C, U, R\}$, thus the spatial cost is

$$\text{SPCost} = \frac{\text{NNZ}(C) + \text{NNZ}(U) + \text{NNZ}(R)}{\text{NNZ}(A)}, \tag{34}$$

   where NNZ(.) presents the number of non-zero entries in a matrix. Other clustering methods usually require large working space in addition to the storage of the original data. Their spatial cost is not evaluated in our experiments.

5. Sparseness: The sparsity of a vector $\mathbf{x}$ is defined as Hoyer (2004),

$$\text{sparseness}(\mathbf{x}) = \frac{\sqrt{d} - (\sum |x_i|)/\left(\sqrt{\sum x_i^2}\right)}{\sqrt{d} - 1}, \tag{35}$$

where d is the dimensionality of $\mathbf{x}$ and x_i is the ith entry of $\mathbf{x}$. Without loss of generality, we define the sparseness of a matrix X as the average sparseness of its columns,

$$\text{sparseness}(X) = \frac{1}{n}\sum_i \text{sparseness}(X(:, i)), \tag{36}$$

where n is the number of columns.

4.2 Synthetic data

In this section, we first provide an illustrative example by taking a small input matrix and carefully examining the clustering results of EMD, i.e., the weight matrix W, the indicator matrix G, the cluster centroids F, and their sparsity. Next, on two overlapping and noisy datasets, we visually compare the interpretability of EMD with two other methods, i.e., CN and Kmeans, whose results can be explained based on the cluster centroids. The sparsity of factors and clustering accuracy are also reported.

4.2.1 An illustrative example

Suppose that we want to approximate and decompose the data matrix $A_{5\times 7}$ as follows,

$$\mathbf{A} = \begin{pmatrix} 13 & 12 & 6 & 12 & -1 & 0 & 0 \\ 5 & 6 & 2 & 7 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 4 & 7 & 4 \\ 0 & 0 & 1 & 0 & 4 & 8 & 8 \\ 0 & 0 & -3 & 0 & 5 & 12 & 13 \end{pmatrix}.$$

By enumerating all possible labeling with the k-means object function,

$$E = \sum_{l=1}^{k} \frac{1}{n_l} \sum_{i \in \mathcal{V}_l} \|X(:, i) - \bar{X}_l\|^2,$$

where k is the number of clusters, n_l is the number of data points of the lth cluster, $\mathcal{V}_l$ is the lth cluster, $\bar{X}_l$ is the mean of the lth cluster, the optimal clustering is: the data in the first four columns are in one cluster, and that in the last three columns are in another, and the minimum k-means error is 32.58. In addition, we add noises to the third and fifth columns. Setting $\alpha = 0.1$ and $c_{max} = 5$, we obtain the data subspaces,

$$\mathbf{C_{EMD-C}} = \begin{pmatrix} 13 & 12 & 0 & 0 \\ 5 & 7 & 0 & 0 \\ 0 & 0 & 7 & 4 \\ 0 & 0 & 8 & 8 \\ 0 & 0 & 12 & 13 \end{pmatrix}, \quad \mathbf{C_{EMD-QR}} = \begin{pmatrix} 0 & 13 & 0 & 12 \\ 0 & 5 & 0 & 7 \\ 7 & 0 & 4 & 0 \\ 8 & 0 & 8 & 0 \\ 12 & 0 & 13 & 0 \end{pmatrix},$$

by EMD-C and EMD-QR, respectively. It is obvious that the columns, i.e.,{1, 4, 6, 7}, are selected without noise to form the representative data subspace. By performing matrix decomposition with EMD-C, EMD-QR, and CN, we get the following weight matrices (all column vectors are normalized),

$$\mathbf{W_{EMD-C}} = \begin{pmatrix} 0.54 & 0.00 \\ 0.46 & 0.00 \\ 0.00 & 0.61 \\ 0.00 & 0.39 \end{pmatrix}, \quad \mathbf{W_{EMD-QR}} = \begin{pmatrix} 0.63 & 0.00 \\ 0.00 & 0.53 \\ 0.37 & 0.00 \\ 0.00 & 0.47 \end{pmatrix}, \mathbf{W_{CN}} = \begin{pmatrix} 0.23 & 0.00 \\ 0.33 & 0.00 \\ 0.16 & 0.00 \\ 0.28 & 0.00 \\ 0.00 & 0.27 \\ 0.00 & 0.35 \\ 0.00 & 0.38 \end{pmatrix}.$$

As described earlier, each entry of the weight matrix represents the weight of an exemplar to a cluster centroid. In W_{CN}, the third and fifth data samples have non-zero weights for clusters one and two, respectively. That is, noisy samples will be included when computing the cluster centroids. On the other hand, each column vector in $W_{\mathrm{EMD-C}}$ and $W_{\mathrm{EMD-QR}}$ has only two non-zero values. So, the cluster centroids by EMD are constructed only based on the selected relevant samples, which makes it robust to noise. The sparseness of weight matrices are comparable between EMD and CN: sparseness($W_{\mathrm{EMD-C}}$) = 0.61, sparseness($W_{\mathrm{EMD-QR}}$) = 0.61, and sparseness(W_{CN}) = 0.50.

Next, we show the cluster indicator matrices:

$$\mathbf{G_{EMD-C}} = \begin{pmatrix} 1.00 & 0.00 \\ 0.97 & 0.00 \\ 0.45 & 0.00 \\ 1.00 & 0.00 \\ 0.00 & 0.47 \\ 0.00 & 1.00 \\ 0.00 & 0.99 \end{pmatrix}, \mathbf{G_{EMD-QR}} = \begin{pmatrix} 0.01 & 1.04 \\ 0.01 & 0.96 \\ 0.00 & 0.46 \\ 0.01 & 0.96 \\ 0.47 & 0.00 \\ 1.03 & 0.00 \\ 0.96 & 0.01 \end{pmatrix}, \mathbf{G_{CN}} = \begin{pmatrix} 1.10 & 0.02 \\ 1.07 & 0.01 \\ 0.51 & 0.00 \\ 1.10 & 0.01 \\ 0.00 & 0.55 \\ 0.04 & 1.18 \\ 0.04 & 1.15 \end{pmatrix},$$

and the corresponding sparseness: sparseness($G^T_{\mathrm{EMD-C}}$) = 1.00, sparseness ($G^T_{\mathrm{EMD-QR}}$) = 0.99, and sparseness(G^T_{CN}) = 0.96.

The cluster centroid matrices are computed as,

$$\mathbf{F_{EMD-C}} = \begin{pmatrix} 12.53 & 0.00 \\ 5.92 & 0.00 \\ 0.00 & 5.84 \\ 0.00 & 8.00 \\ 0.00 & 12.38 \end{pmatrix}, \mathbf{F_{EMD-QR}} = \begin{pmatrix} 0.00 & 12.53 \\ 0.00 & 5.93 \\ 5.89 & 0.00 \\ 8.00 & 0.00 \\ 12.36 & 0.00 \end{pmatrix},$$

$$\mathbf{F_{CN}} = \begin{pmatrix} 11.29 & -0.27 \\ 5.38 & -0.55 \\ 0.15 & 5.01 \\ 0.15 & 6.89 \\ -0.47 & 10.46 \end{pmatrix}, \mathbf{F_{Kmeans}} = \begin{pmatrix} 10.75 & -0.33 \\ 5.00 & -0.66 \\ 0.25 & 5.00 \\ 0.25 & 6.66 \\ -0.75 & 10.00 \end{pmatrix}.$$

Clearly, EMD generates a more meaningful cluster centroid matrix. In the first column vector of $\mathbf{F_{EMD-C}}$ and the second column vector of $\mathbf{F_{EMD-QR}}$, only the top two elements have non-zero values, implying these two features are used for the first cluster only. Similar conclusion can be drawn for the second cluster. This owes to the exemplar selection, which effectively removes the noisy data. On the other hand, by using all data points, the centroid matrices for CN and Kmeans have no zero entries, meaning every feature is employed in the clustering.

4.2.2 Interpretability

In the following, we evaluate EMD on two larger synthetic datasets. The first one (Dataset I) has two clusters, while the second one (Dataset II) has three clusters. Each cluster contains 100 samples drawn from a multivariate gaussian distribution $\mathcal{N}(\mu_i, \Sigma_i)$ $(i = 1, 2, 3)$ with means ranging from -0.5 to 0.5. For simplicity, we restrict the covariance matrices to be diagonal with variance ranging from 1 to 11. After generating clusters, we add 20 % random value noise, ranging from 0 to 1, to both datasets. In our experiment, we set $d = 400$, and generate a 400×200 matrix for Dataset I and a 400×300 matrix for Dataset II. For the two datasets, we specify $\alpha = 0.3$ and $c_{max} = n$ (n is the number of data points). To visualize data clusters and the centroids, we reduce the dimensions from 400 to 2 using PCA and then display data in a two-axis layout. In Fig. 1, data distributions are shown. Clearly, both datasets are overlapping and noisy.

In Fig. 1, the cluster centroids computed by EMD-C, EMD-QR, CN and Kmeans are denoted by red stars, dark green dots, light green triangles, and brown diamonds, respectively. As shown, EMD-C gains the desirable cluster centroids on both datasets, which are clearly located around the centers of the clusters. EMD-QR and CN also gain the comparable results on Dataset I, while on Dataset II, the cluster centroids are slightly deviated from the cluster centers. Kmeans performs poorly on both datasets: the cluster centroids are all located in heavy overlapping and noisy areas. In Table 2, we compared the clustering accuracy among the four algorithms. Clearly, EMD-C achieves the highest accuracy on both datasets. EMD-QR and CN obtain the comparable results, while Kmeans gets the lowest values. The high accuracy of EMD clearly shows that it is very robust to noise. In addition, we compare the sparseness of factors between EMD and CN in Table 3. EMD-C and EMD-QR achieve the comparable sparseness of weight and indicator matrices with CN: EMD gains a slightly lower sparseness of the weight matrices than CN, while getting higher values for the indicator matrices.

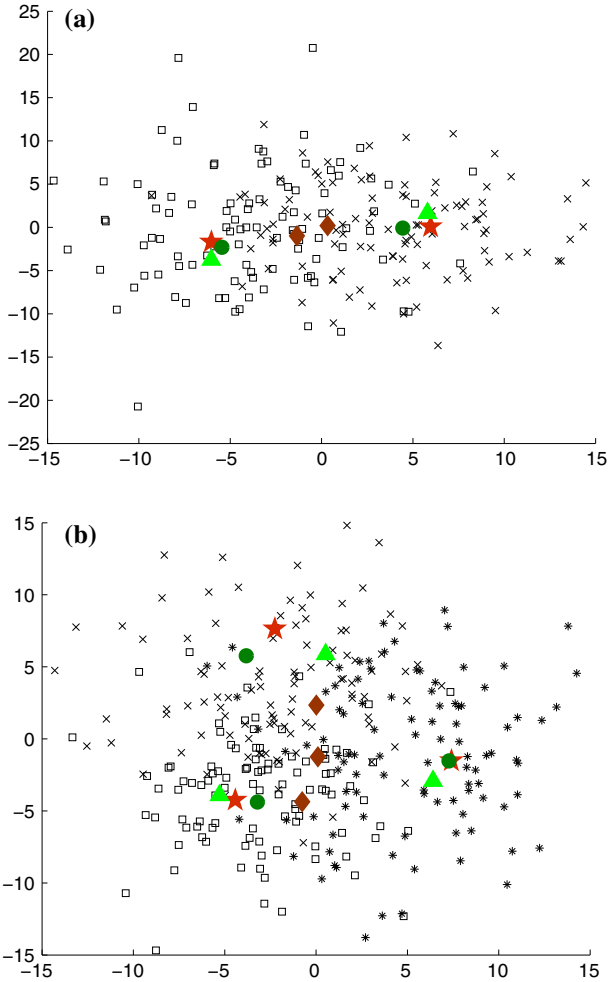

Fig. 1 Comparison of cluster centroids gained by EMD-C, EMD-QR, CN and Kmeans. *Big star* presents the centroid gained by EMD-C, *filled circle* the centroid by EMD-QR, *open triangle* the centroid gained by CN, *open diamond* the centroid gained by Kmeans. **a** Shows the results for Dataset I, and **b** shows the results for Dataset II

4.3 Real data

We have evaluated six clustering algorithms, namely, EMD-C, EMD-QR, CN, NMF, CoNMF, Kmeans and SVDC, on several real-world datasets. When clustering real-world data, the number of clusters is typically unknown and thus has to be estimated through model detection. Some well-known approaches on this topic include Bayesian inference criteria (BIC) Schwarz (1978) and minimum description length (MDL) Barron et al. (1998). In our experiments, we choose the datasets publicly available with known class labels and set the number of clusters based on the ground truth.

Table 2 Clustering accuracy on two synthetic datasets

Method	Dataset I	Dataset II
EMD-C	0.7650	0.7067
EMD-QR	0.7300	0.6433
CN	0.7350	0.6867
Kmeans	0.5500	0.4100

Table 3 Sparseness of factors

Sparseness of W		
Method	Dataset I	Dataset II
EMD-C	0.41	0.52
EMD-QR	0.42	0.52
CN	0.42	0.53
Sparseness of G		
Method	Dataset I	Dataset II
EMD-C	0.77	0.71
EMD-QR	0.80	0.72
CN	0.74	0.69

4.3.1 Data description

The first real-world dataset we used is 20 *Newsgroups* Lang (1995) [1], a collection of approximately 20, 000 messages from UseNet news, partitioned (nearly) evenly across 20 different newsgroups. To thoroughly test the proposed methods, we mix and construct three datasets, having classes ranging from 4 to 20 and data samples from about 4000 to 19, 000. Reuters-21578, Distribution 1.0 [2] is a collection of documents from Reuters newswire, which contains 21578 documents from 135 topics. We remove the documents with multiple category labels, as well as the categories with less than 100 documents, and finally get a dataset with 8608 documents consisting of 20 categories. In our experiments, we use a few subsets of this collection with the clusters ranging from 10 to 20 and data samples from 500 to 8608. Another text corpus we used is WebKB [3], a WWW-pages collection from computer science departments of various universities. We process each web page as a document and mainly use 4991 web pages with six categories in the experiment. In addition, we performed experiments on three large datasets, collected from LIBSVM Data. [4] One is the MNIST database of handwritten digits, which contains about 76, 000 examples. The digits have been size-normalized and centered in a fixed-size image. Another is Real-Sim, which contains about 72, 000 UseNet articles from four discussion groups, for simulated auto racing, simulated aviation, real autos and real aviation. In our experiments, we use this data for

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/.

[2] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html.

[3] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/.

[4] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.

Table 4 Real-world datasets

Name	Data structure	No. of clusters	Size of data ($d \times n$)
Newsgroup4	comp.graphics, rec.sport.baseball, sci.crypt, sci.med	4	$6,163 \times 3,948$
Newsgroup8	rec.autos, rec.motorcycles, rec.baseball, rec.hockey sci.crypt, sci.electronics, sci.med, sci.space	8	$6,163 \times 7,931$
Newsgroup20	alt.atheism, comp.graphics, comp.misc, mac.hardware pc.hardware, windows.x, misc.forsale, rec.autos rec.motorcycles, rec.baseball, sport.hockey, sci.crypt sci.electronics, sci.med, sci.space, soc.christian talk.guns, talk.mideast, politics.misc, religion.misc	20	$6,163 \times 18,846$
Reuters10	trade, ship, acq, earn, sugar money-fx, interest, coffee, crude, money-supply	10	$19,418 \times 8,024$
Reuters20	trade, grain, ship, gold, acq ipi, earn, jobs, sugar,cpi money-fx, interest, cocoa, coffee, crude money-supply, copper, alum, reserves, gnp	20	$19,418 \times 8,608$
WebKB	student, faculty, course, project, staff, department	6	$1,938 \times 4,991$
MNIST	"0","1","2","3","4" "5","6","7","8","9"	10	$784 \times 76,054$
Real-Sim	real, simulated	2	$2,011 \times 72,201$
Webspam	positive, negative	2	$128 \times 154,123$

Table 5 Six datasets from Reuters10 with 10 clusters

Datasets	D1	D2	D3	D4	D5	D6
n	500	1,000	1,428	1,729	2,029	2,301

a two-cluster partition: the real and the simulated. The last is Webspam, a collection of web pages that are created to manipulate search engines and deceive Web users, including positive and negative examples. For all the text collections, the common words are removed, and the meaningful words are stemmed using Porter's suffix-stripping algorithm Porter (1980). Table 4 shows the detailed description of all the datasets.

4.3.2 Effects on α and c

First, we study the performance of EMD with respect to the choice of α, the tolerance level for the low-rank approximation. We have constructed six datasets from Reuters10, each having ten clusters with data samples ranging from 500 to 2, 301 (see Table 5).

In Figs. 2 and 3, we plot the average clustering accuracy across six datasets against α for EMD-C and EMD-QR, respectively, where the standard deviation is shown as the bars. As shown, both the NMI and Ac values increase quickly as α decreases until it reaches about 0.3, after which, when α continues to decrease, the curves of NMI and Ac

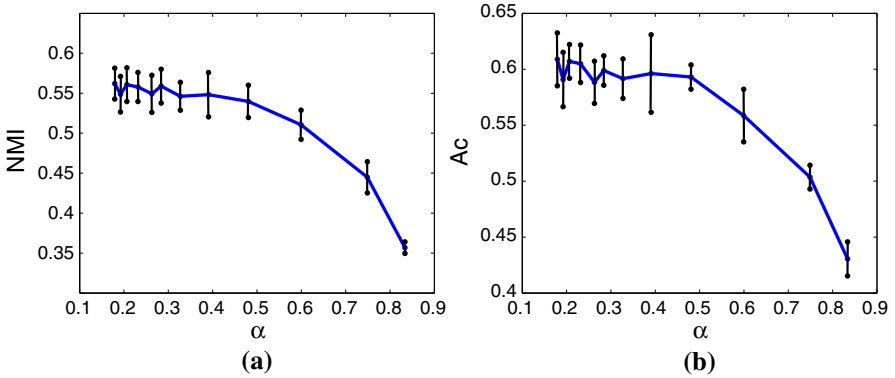

Fig. 2 Effects of tolerance ratio α for EMD-C. **a** Shows α versus NMI, and **b** shows α versus Ac. *Bars* show the standard deviation over six datasets

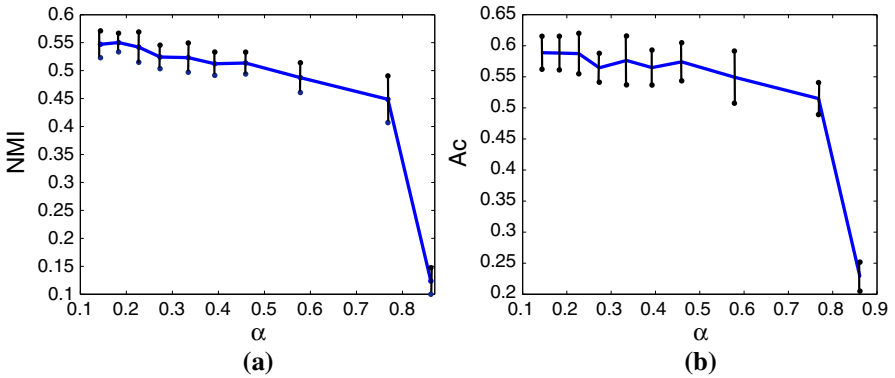

Fig. 3 Effects of tolerance ratio α for EMD-QR. **a** Shows α versus NMI, and **b** shows α versus Ac. *Bars* show the standard deviation over six datasets

become flat with fluctuations. This indicates that EMD gains better performance as α decreases until a certain value. After that, further reduction of α provides no significant improvement on the clustering accuracy. Based on these results, we empirically set α at 0.3 in all our following experiments.

As α is changed, EMD gains subspaces of different sizes accordingly. Since the selection of subspaces is a very important step in EMD, we next show how the subspace size affects the clustering performance. In Figs. 4 and 5, we plot the clustering accuracy against the subspace size c on all the datasets for both EMD-C and EMD-QR. As shown, starting with a small and incomplete subspace, EMD performs poorly. When more exemplars are added, the subspace will include more bases of the data, so the performance becomes better. However, when c increases over a certain point, e.g., $c = 150$ on D1 and $c = 110$ on D2 in Figs. 4a and 5a, noise will be included in the subspace, resulting in unstable clustering accuracy. The results for EMD-QR show the similar trends.

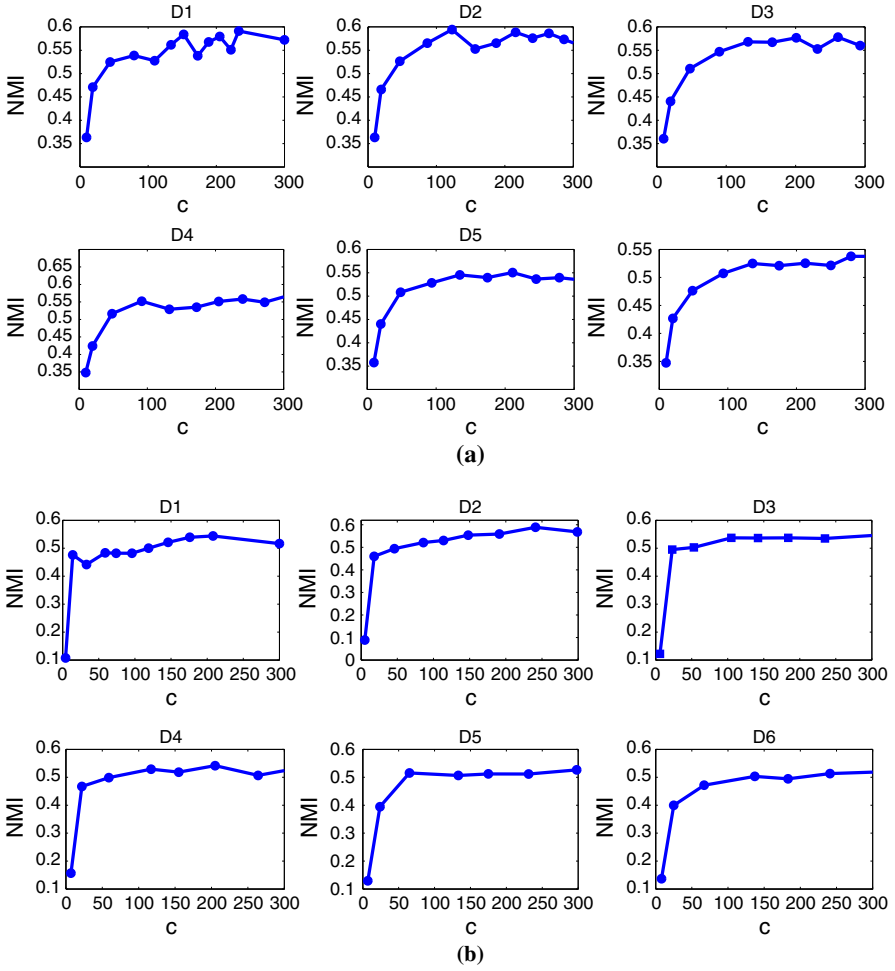

Fig. 4 Effects of the subspace size c on NMI. **a** c versus NMI on six datasets for EMD-C, **b** c versus NMI on six datasets for EMD-QR

We also show the running time and spatial costs associated with various c. In Fig. 6, we plot the running time against c for all the datasets. Clearly, all the curves climb with the increase of c, and EMD constantly takes less time on smaller datasets than on larger ones. This is consistent with our analysis on the computation complexity of EMD. In Fig. 7, we show the spacial costs with varying values of c. First, both methods require more space as c increases. In addition, given the same subspace size, EMD-QR generally requires less space than EMD-C. For example, with $c = 150$, the spacial cost for EMD-QR is less than 1 on most datasets, while it is as much as 3 for EMD-C. This is mainly because the 844 algorithm selects columns and rows to construct C and R, both of which reserve the sparsity of the original data. On the other hand, *Colibri* only selects C, and constructs a dense R using $R = C^T A$, resulting in a higher spatial cost.

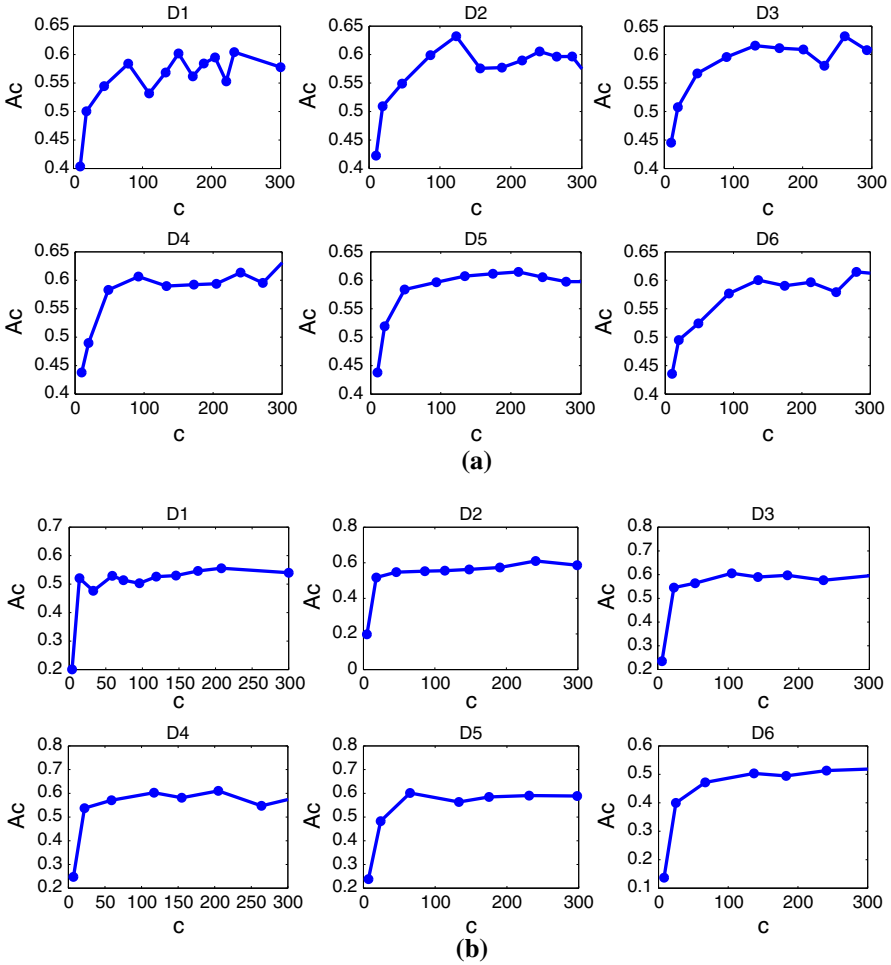

Fig. 5 Effects of the subspace size c on Ac. **a** c versus Ac on six datasets for EMD-C, **b** c versus Ac on six datasets for EMD-QR

Based on these results, we fix $\alpha = 0.3$ and $c_{max} = 300$, and compare EMD with other clustering methods on the clustering accuracy and running time. As shown in Fig. 8, EMD achieves the highest clustering accuracy on most datasets; NMF, CN, and CoNMF gain competitive results; and Kmeans and SVDC perform poorly. The running time is shown in Fig. 9 by plotting time vs. n across six datasets. Clearly, Kmeans is the fastest, followed by EMD-C, SVDC, EMD-QR, and NMF. CoNMF is much slower than those five algorithms: on average six times slower than EMD. CN needs even more time, dozens more than EMD. Note the different scales we used to draw the top and bottom half of Fig. 9.

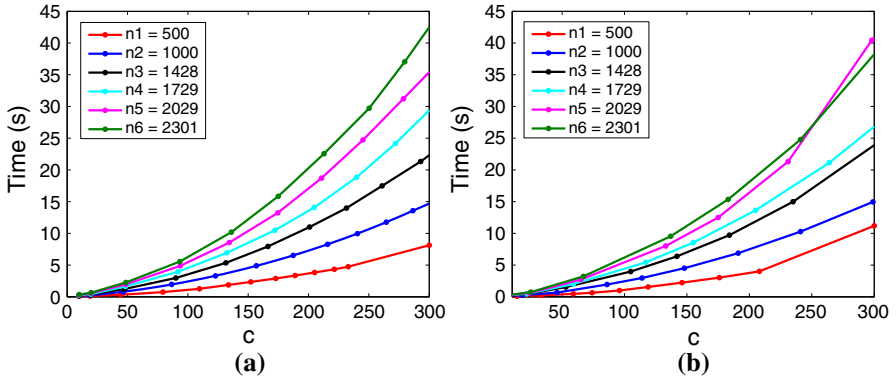

Fig. 6 Effects of the subspace size c on the running time. **a** Shows c versus time for EMD-C, and **b** shows c versus time for EMD-QR

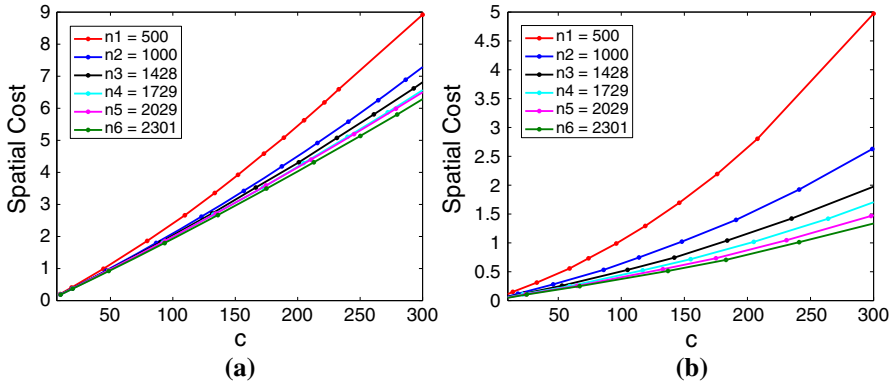

Fig. 7 Effects of the subspace size c on spacial costs. **a** Shows c versus spatial costs for EMD-C, and **b** shows c versus spatial costs for EMD-QR

4.3.3 Experimental results

Finally, we evaluate the performance of EMD on all real-world datasets with $\alpha = 0.3$ and $c_{max} = 500$. Since CN becomes extremely slow when n is large, we only compare EMD with NMF, CoNMF, Kmeans and SVDC in the following experiments.

In Table 6, the experimental results on NMI, Ac, the running time and spatial cost are reported. Clearly, among all six methods, EMD-C gains the highest clustering accuracy on most of the datasets. Particularly, EMD-C falls shortly behind NMF and Kmeans on Newsgroup8 and MNIST, respectively. Notice that NMF gains competitive results with EMD-C on some datasets, such as Newsgroup and Reuters. These are very complex datasets with high sparsity, large dimensionality, and many categories. Consequently, insufficient or noisy sampling (noisy data with large norms) may lead to poor low-rank matrix approximation as well as unsatisfactory clustering results.

Notice that EMD-QR gains comparable accuracy with EMD-C on most text datasets except for Newsgroup20, where it falls behind EMD-C and NMF. On MNIST, EMD-

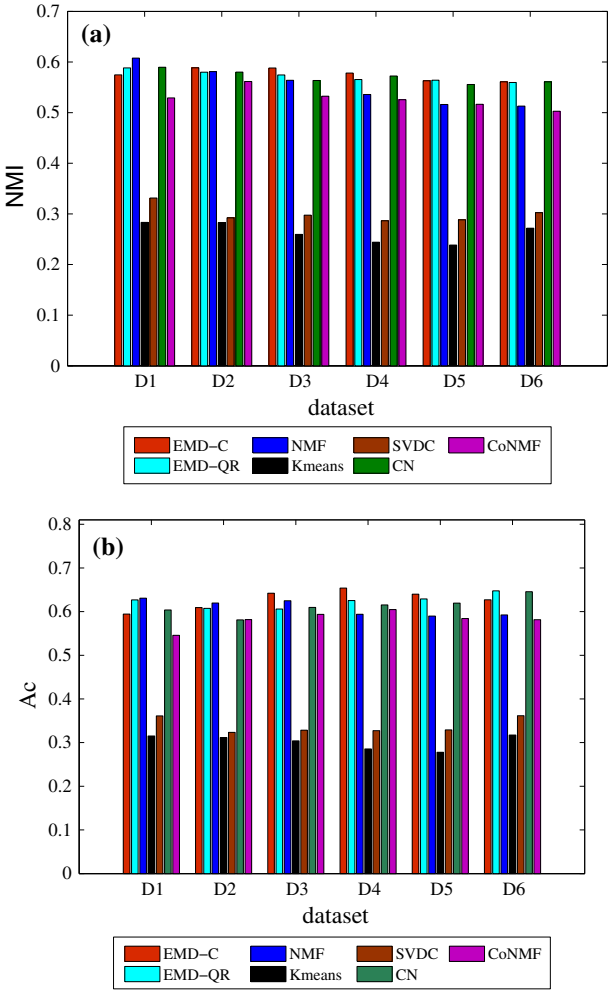

Fig. 8 Comparisons of clustering accuracy among EMD-C, EMD-QR, NMF, Kmeans, SVDC, CN, and CoNMF. **a** Shows NMI on six datasets, and **b** shows Ac on six datasets

QR performs worse. This is mainly due to the worse quality of the low-rank approximation. We also observe that EMD performs poorly on Webspam. This is partly due to selecting a very small amount of samples, determined by the available memory space, and they are not enough to cover the whole spectrum of the data. As discussed in Sect. 3.1, the approximate subspace may be used to sample sufficient data points to achieve accurate matrix approximation as well as clustering results. We employ *Colibri* Tong et al. (2008) to compute $\tilde{R}$, and then use it to compute W and G in EMD-QR. Compared with 27 samples selected in the original EMD-QR, we now can select 56 data points with $\tilde{R}$. Consequently, the clustering accuracy is increased to 0.7004 (highest among the algorithms). We find out that further increase of data samples will

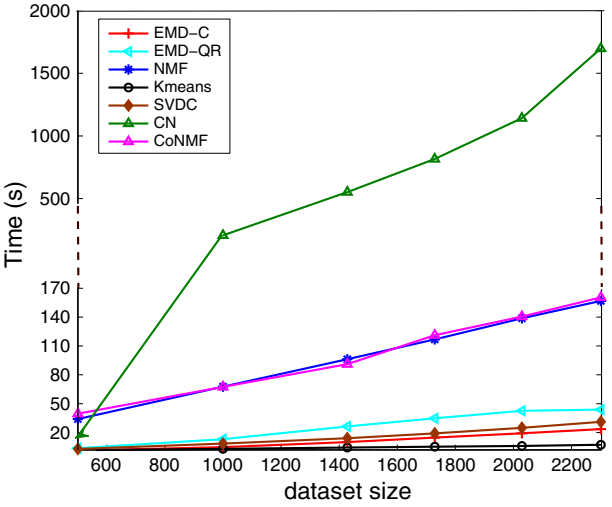

Fig. 9 Comparisons of the running time among EMD-C, EMD-QR, NMF, Kmeans, SVDC, CN, and CoNMF on six datasets

not help improve the accuracy due to the complexity of intrinsic structure of this data set.

CoNMF typically achieves the comparable results with NMF on all the datasets. Notice that no result is reported for CoNMF on Reuters20. This is due to the large number of categories (i.e., 20) to be clustered for both columns and rows, for which CoNMF requires much more space than the total memory of the PC used in our experiment. Kmeans and SVDC have very low clustering accuracy on most datasets except on MNIST. This shows that they are not suitable to handle high-dimensional datasets, commonly found in text corpus.

For computational speed, the actual running time for each algorithm is reported. Kmeans generally runs fastest among the algorithms, followed by SVDC, which requires extra time for the SVD procedure. Among all matrix decomposition-based clusterings, EMD-C runs fastest. Clearly, EMD-C is efficient to process large datasets, e.g., requiring < 200 s for the MNIST dataset. In particular, it runs very fast on the datasets when a small subspace is needed, i.e., in Newsgroup4, Newsgroup8, WebKB, and Webspam. Compared with EMD-C, EMD-QR runs slower since it uses a larger subspace for the given error tolerance. NMF runs slow, especially on high-dimensional datasets, e.g., Newsgroup20, Reuters10 and Reuters20. CoNMF is typically slower than NMF, due to computing more factors.

The spatial cost is highly related with the sparsity of the data, i.e., the more sparse the data is, the lower the spatial cost (more space will be saved). In addition, the number of exemplars affects the spatial cost: less space is needed if less exemplar is selected. Thus, the spatial cost is not directly related to the size of the datasets. From Table 6, we notice that EMD-QR typically has a cost less than one on all the datasets except for Reuters10 and Reuters20, where the costs are higher with more exemplars. On sparse datasets such as newsgroup, WebKB, Real-sim, and Webspam,

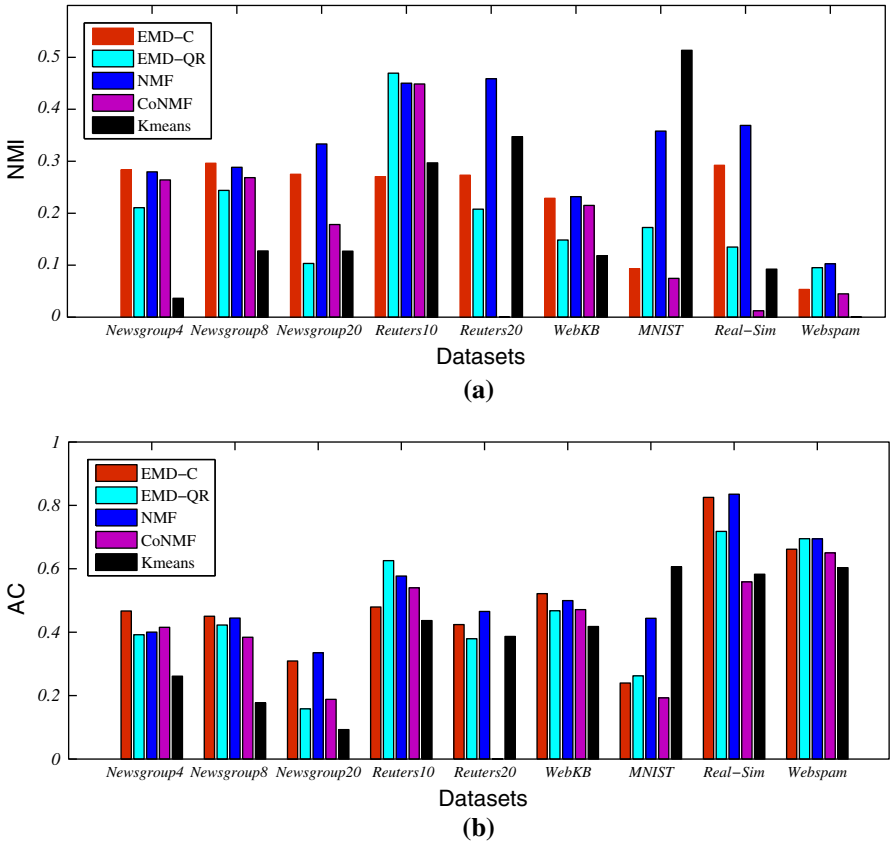

Fig. 10 Comparisons among EMD-C, EMD-QR, NMF, CoNMF, Kmeans on nine datasets in fixed time (300 s). **a** datasets versus NMI, **b** datasets versus Ac

the spatial cost is relatively small. Besides, the spatial cost is small on MNIST as well due to selecting less number of exemplars. EMD-C generally requests more space than EMD-QR. This is mainly due to the large storage requirement of the dense matrix R, which is computed from the selected columns.

The convergence speed is different among EMD and other algorithms with iterative optimization. As an alternative comparison, we evaluate these methods with a running time limit at 300 s, which is sufficient to produce meaningful results on most datasets. Figure 10 shows the comparisons of NMI and Ac among EMD-C, EMD-QR, NMF, CoNMF and Kmeans. As seen, the results are similar to what we reported in Table 6. NMF is the simplest iterative method among all the algorithms we compared. It needs fewer repetitions to converge, but runs slowly in each repetition when using the original matrix. On the other hand, EMD leverages low-rank matrix approximation and provides great computational and spatial efficiency. In addition, EMD clustering results are sparse and easy for interpretation. Kmeans typically performs poorly on most datasets.

Table 6 Experimental results on large real-world datasets

Datasets	$d \times n$	k	Method (c)	NMI	Ac	Running time (s)	Space
Newsgroup4	6,163 × 3,948	4	EMD-C (71)	0.2753	0.4085	5.97	1.10
			EMD-QR (197)	0.1955	0.389	48.78	0.56
			NMF	0.2703	0.3986	37.72	–
			CoNMF	0.2671	0.4167	49.57	–
			Kmeans	0.0288	0.2593	5.55	–
			SVDC	0.0249	0.2587	32.41	–
Newsgroup8	6,163 × 7,931	8	EMD-C (96)	0.2762	0.4197	23.82	1.42
			EMD-QR (346)	0.2013	0.3659	313.01	0.67
			NMF	0.2773	0.4267	77.95	–
			CoNMF	0.2685	0.3495	73.75	–
			Kmeans	0.0757	0.15	12.00	–
			SVDC	0.0675	0.1507	53.14	–
Newsgroup20	6,163 × 18,846	20	EMD-C (118)	0.3088	0.3268	76.50	1.72
			EMD-QR (500)	0.2261	0.2626	1078.15	0.64
			NMF	0.3108	0.3101	320.35	–
			CoNMF	0.3060	0.2901	240.93	–
			Kmeans	0.1111	0.0893	40.33	–
			SVDC	0.1242	0.093	55.53	–
Reuters10	19,418 × 8,024	10	EMD-C (281)	0.4269	0.4969	135.33	7.09
			EMD-QR (468)	0.4285	0.5103	474.52	1.12
			NMF	0.4198	0.4968	461.05	–
			CoNMF	0.3881	0.5034	411.61	–
			Kmeans	0.2883	0.4216	22.35	–
			SVDC	0.3005	0.4097	53.92	–
Reuters20	19,418 × 8,608	20	EMD-C (356)	0.4149	0.3834	220.75	8.75
			EMD-QR (500)	0.4049	0.3697	442.55	1.54
			NMF	0.4273	0.3655	728.98	–
			CoNMF	–	–	–	–
			Kmeans	0.3508	0.3424	32.21	–
			SVDC	0.3469	0.3551	57.17	–
WebKB	1,938 × 4,991	6	EMD-C (94)	0.2124	0.4763	13.42	1.26
			EMD-QR (213)	0.1219	0.4228	20.86	0.27
			NMF	0.1986	0.4578	32.87	–
			CoNMF	0.2085	0.4247	39.06	–
			Kmeans	0.0881	0.3946	6.29	–
			SVDC	0.1041	0.4051	32.04	–

Table 6 continued

Datasets	$d \times n$	k	Method (c)	NMI	Ac	Running time (s)	Space
MNIST	$784 \times 76{,}054$	10	EMD-C (50)	0.3442	0.4414	102.76	0.31
			EMD-QR (50)	0.2506	0.332	172.54	0.17
			NMF	0.3437	0.4044	198.73	–
			CoNMF	0.3082	0.3745	218.33	–
			Kmeans	0.4579	0.5258	221.77	–
			SVDC	0.4514	0.505	45.29	–
Real-sim	$2{,}011 \times 72{,}201$	2	EMD-C (100)	0.2809	0.7902	166.49	1.96
			EMD-QR (262)	0.1652	0.7444	276.29	0.15
			NMF	0.2825	0.7296	255.53	–
			CoNMF	0.1598	0.5500	335.39	–
			Kmeans	0.053	0.6573	37.35	–
			SVDC	0.0246	0.6574	54.84	–
Webspam	$128 \times 154{,}123$	2	EMD-C (5)	0.0536	0.6616	18.04	0.5056
			EMD-QR (27)	0.1139	0.6964	73.28	0.047
			NMF	0.1165	0.6958	32.36	–
			CoNMF	0.0953	0.6929	144.66	–
			Kmeans	0.000095	0.6033	295.06	–
			SVDC	0.1104	0.6968	49.08	–

The subspace size c for EMD-C and EMD-QR is reported as the value in the parenthesis behind the corresponding methods. A dash "–" denotes "not evaluated"

5 Conclusion

In the paper, we present EMD, a theoretical framework to cluster large scale datasets. By uniquely combining matrix decomposition-based clustering and low-rank matrix approximation, EMD has several advantages over existing clustering methods: (1) It is robust to noise through exemplar selection in matrix approximation, leading to higher clustering accuracy; (2) It gains efficiency in both computational time and space by decomposing the compact approximation; (3) It guarantees the basis matrix to lie within the representative subspace, making cluster centroids more interpretable; and (4) It tends to generate sparse factors, and thus a sharp partition of the data. From a theoretical perspective, we mathematically show the correctness and convergence of EMD, and provide detailed analysis on its computational efficiency. Empirically, we demonstrate the performance of EMD through extensive experiments on both synthetic and real-world data.

References

Achlioptas D, Mcsherry F (2007) Fast computation of low-rank matrix approximations. J ACM 54(2):9
Baker CTH (1997) The numerical treatment of integral equations. Clarendon Press, Oxford

Barron AR, Rissanen J, Yu B (1998) The minimum description length principle in coding and modeling. IEEE Trans Inf Theory 44(6):2743–2760
Berry MW, Browne M, Langville AN, Pauca PV, Plemmons RJ (September 2007) Algorithms and applications for approximate nonnegative matrix factorization. Comput Stat Data Anal 52(1):155–173
Berry MW, Pulatova SA, Stewart GW (2005) Algorithm 844: computing sparse reduced-rank approximations to sparse matrices. ACM Trans Math Softw 31(2):252–269
Chen Y, Wang L, Dong M, Hua J (2009) Exemplar-based visualization of large document corpus. IEEE Trans Visual Comput Graphics 15(6):1169–1176
Chung FRK (1997) Spectral graph theory. American Mathematical Society
Delves LM, Mohamed JL (1985) Computational methods for integral equations. Cambridge University Press, New York
Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the em algorithm. J R Stat Soc B 39:1–38
Dhillon I, Guan Y, Kulis B (2004) Kernel k-means: spectral clustering and normalized cuts. In: Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 551–556
Dhillon IS, Guan Y, Kulis B (2005) A unified view of kernel k-means, spectral clustering and graph cuts. Technical Report TR-04-25, University of Texas Dept. of Computer Science
Ding C, He X, Simon HD (2005) On the equivalence of nonnegative matrix factorization and spectral clustering. In: Proceedings of SIAM International Conference of Data Mining, pp 606–610
Ding C, He X, Zha H, Simon HD (2001) A min-max cut algorithm for graph partitioning and data clustering. In: IEEE International Conference on Data Mining, pp 107–114
Ding C, Li T, Jordan MI (2008) Convex and semi-nonnegative matrix factorizations. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 99. IEEE Computer Society, Los Alamitos
Ding C, Li T, Peng W (2006) Nonnegative matrix factorization and probabilistic latent semantic indexing: equivalence chi-square statistic, and a hybrid method. Proc Natl Conf Artif Intell 21(1):342
Ding C, Li T, Peng W, Park H (2006) Orthogonal nonnegative matrix t-factorizations for clustering. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 126–135
Drineas P, Frieze A, Kannan R, Vempala S, Vinay V (2004) Clustering large graphs via the singular value decomposition. IEEE J Mach Learn 56(1–3):9–33
Drineas P, Kannan R, Mahoney M (2006) Fast monte-carlo algorithms for matrices ii: computing low-rank approximations to a matrix. SIAM J Comput 36:158–183
Drineas P, Kannan R, Mahoney MW (2006) Fast monte carlo algorithms for matrices iii: computing a compressed approximate matrix decomposition. SIAM J Comput 36:184–206
Drineas P, Mahoney MW (2005) On the nyström method for approximating a gram matrix for improved kernel-based learning. J Mach Learn Res 6:2153–2175
Duda HO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, New York
Fiedler M (1973) Algebraic connectivity of graphs. Czechoslov Math J 23(98):298–305
Fowlkes C, Belongie S, Chung F, Malik J (2004) Spectral grouping using the nyström method. IEEE Trans Pattern Anal Mach Intell 26(2):214–225
Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman, New York
Golub GH, Van Loan CF (1996) Matrix computations, 3rd edn. Johns Hopkins University Press, Baltimore
Hagen L, Kahng AB (1992) New spectral methods for ratio cut partitioning and clustering. IEEE Trans Comput Aided Des Integr Circuits Syst 11(9):1074–1085
Hoyer PO (2004) Non-negative matrix factorization with sparseness constraints. J Mach Learn Res 5:1457–1469
Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. ACM Comput Surv 31:264–323
Jolliffe IT (2002) Principal component analysis, 2nd edn. Springer, New York
Kim H, Park H (2007) Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics 23(12):1495–1502
Lang K (1995) News weeder: learning to filter netnews. In: Proceedings of the 12th International Conference on Machine Learning, pp 331–339
Lee DD, Seung HS (1999) Learning the parts of objects by non-negative matrix factorization. Nature 401(6755):788–791
Lee DD, Seung HS (2000) Algorithms for non-negative matrix factorization. Neural Inf Proc Syst 13:556–562

Li T, Ding C (2006) The relationships among various nonnegative matrix factorization methods for clustering. In: Proceedings of the IEEE International Conference on Data Mining, pp 362–371
MacQueen JB (1967) Some methods for classsification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, pp 281–297
Mahdavi M, Abolhassani H (2009) Harmony k-means algorithm for document clustering. Data Min Knowl Disc 18:370–391. doi:10.1007/s10618-008-0123-0
Porter MF (1980) An algorithm for suffix stripping. Program 14(3):130–137
Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6(2):461–464
Sha F, Lin Y, Saul LK, Lee DD (2007) Multiplicative updates for nonnegative quadratic programming. Neural Comput 19(8):2004–2031
Shi J, Malik J (2000) Normalized cuts and image segmentation. IEEE Trans Pattern Anal Mach Intell 22(8):888–905
Shyamalkumar ND, Varadarajan K (2007) Efficient subspace approximation algorithms. In: SODA '07 Proceedings of the 18th annual ACM-SIAM symposium on Discrete algorithms, pp 532–540
Stewart GW (1999) Four algorithms for the efficient computation of truncated qr approximations to a sparse matrix. Numer Math 83:313–323
Strehl A, Ghosh J, Cardie C (2002) Cluster ensembles: a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617
Sun J, Xie Y, Zhang H, Faloutsos C (2008) Less is more: sparse graph mining with compact matrix decomposition. Stat Anal Data Min 1(1):6–22
Tong H, Papadimitriou S, Sun J, Yu PS, Faloutsos C (2008) Colibri: fast mining of large static and dynamic graphs. In: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 686–694
Wang F, Li T, Wang X, Zhu S, Ding C (2011) Community discovery using nonnegative matrix factorization. Data Min Knowl Disc 22:493–521. doi:10.1007/s10618-010-0181-y
Wang L, Dong M (2011) On the clustering of large-scale data: a matrix-based approach. In: To appear Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 2011), p 10
Williams CK, Seeger M (2001) Using the nyström method to speed up kernel machines. In: Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference, MIT Press, pp 682–688
Xu W, Gong Y (2004) Document clustering by concept factorization. In: SIGIR '04: proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, New York, pp 202–209
Xu W, Liu X, Gong Y (2003) Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval, pp 267–273
Yan D, Huang L, Jordan M (2009) Fast approximate spectral clustering. Technical Report UCB/EECS-2009-45, EECS Department, University of California, Berkeley
Yen GG, Wu Z (2008) Ranked centroid projection: a data visualization approach with self-organizing maps. IEEE Trans Neural Netw 19(2):245–259
Zhang K, Kwok JT (2006) Block-quantized kernel matrix for fast spectral embedding. In: ICML '06: proceedings of the 23rd international conference on Machine learning, ACM, New York, pp 1097–1104
Zhang K, Tsang IW, Kwok JT (2008) Improved nyström low-rank approximation and error analysis. In ICML '08: proceedings of the 25th international conference on Machine learning, ACM, New York, pp 1232–1239