

SymNMF: nonnegative low-rank approximation of a similarity matrix for graph clustering

Da Kuang · Sangwoon Yun · Haesun Park

Received: 27 January 2014 / Accepted: 20 October 2014 / Published online: 19 November 2014
© Springer Science+Business Media New York 2014

Abstract Nonnegative matrix factorization (NMF) provides a lower rank approximation of a matrix by a product of two nonnegative factors. NMF has been shown to produce clustering results that are often superior to those by other methods such as K-means. In this paper, we provide further interpretation of NMF as a clustering method and study an extended formulation for graph clustering called Symmetric NMF (SymNMF). In contrast to NMF that takes a data matrix as an input, SymNMF takes a nonnegative similarity matrix as an input, and a symmetric nonnegative lower rank approximation is computed. We show that SymNMF is related to spectral clustering, justify SymNMF as a general graph clustering method, and discuss the strengths and shortcomings of SymNMF and spectral clustering. We propose two optimization algorithms for SymNMF and discuss their convergence properties and computational efficiencies. Our experiments on document clustering, image clustering, and image segmentation support SymNMF as a graph clustering method that captures latent linear and nonlinear relationships in the data.

Keywords Symmetric nonnegative matrix factorization · Low-rank approximation · Graph clustering · Spectral clustering

D. Kuang · H. Park (✉)
School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta,
GA 30332-0765, USA
e-mail: hpark@cc.gatech.edu

D. Kuang
e-mail: da.kuang@cc.gatech.edu

S. Yun
Department of Mathematics Education, Sungkyunkwan University, Jongro-gu, Seoul 110-745, Korea
e-mail: yswmathedu@skku.edu

S. Yun
Korea Institute for Advanced Study, Dongdaemun-gu, Seoul 130-722, Korea

1 Introduction

Dimension reduction and clustering are two of the key tasks in machine learning and data analytics. Suppose a collection of n data items with m features is represented in a matrix $X \in \mathbb{R}^{m \times n}$. In a low rank approximation, we are given a desired reduced rank k which is typically much smaller than m and n , and we are to find $C \in \mathbb{R}^{m \times k}$ and $G \in \mathbb{R}^{n \times k}$ such that the difference between X and the product CG^T is minimized as

$$X \approx CG^T. \quad (1)$$

This minimization problem can be formulated using various difference or distance measures. In this paper, we will focus on the Frobenius norm based minimization, i.e.

$$\min_{C,G} \|X - CG^T\|_F. \quad (2)$$

In nonnegative Matrix factorization (NMF), nonnegativity is imposed on the factors C and G , i.e. we are to solve

$$\min_{C \geq 0, G \geq 0} \|X - CG^T\|_F^2, \quad (3)$$

where $C \in \mathbb{R}_+^{m \times k}$, $G \in \mathbb{R}_+^{n \times k}$, and \mathbb{R}_+ denotes the set of nonnegative real numbers. The NMF can be defined for any matrix, but it makes more sense to consider NMF when the matrix X is nonnegative. Throughout this paper, we will assume that $X \in \mathbb{R}_+^{m \times n}$ is nonnegative.

NMF has been shown to be an effective method in numerous applications [11, 20, 25, 29, 35]. In this paper, we will focus on the role of NMF as a clustering method. Note that NMF is posed as a constrained low rank approximation method, and accordingly, is a method for dimension reduction. However, dimension reduction and clustering are closely related. The following interpretation of the results of the low rank approximation illustrates this point: We consider the columns of C are the new basis for the reduced k -dimensional space for X , and each column of G^T provides the k -dimensional representation of the corresponding column of X in the space spanned by the columns of C .

In the case of singular value decomposition (SVD), the columns of C are ordered in a way that the first column is the most dominant vector (the leading left singular vector) that captures the largest variation in the data, and the next column is the second most dominant vector and orthogonal to the leading singular vector, etc. Therefore, the columns of C do not “equally” represent the column space spanned by the data matrix X . In addition, the two factors C and G^T can have negative elements, and thus it will be difficult to interpret the i -th column of G^T as a “proportion distribution” with which the i -th data item has the component in the corresponding basis vector in C . On the other hand, the columns of C in NMF cannot have negative signs, and accordingly cannot “cancel out” some directions that the more dominant columns of C may represent. Accordingly, the columns of C more or less “equally” represent the data set and each column in the factor G^T can be viewed as a distribution with which the i -th data item has the component in the corresponding column of C . Since we can use G^T to derive an assignment of the n data points into k groups, clustering can be viewed as a special type of dimension reduction. The NMF gives a soft clustering result as explained above but we can also interpret the result as a hard clustering by assigning the i -th data point to the j -th cluster when the largest element among all components of the i -th column of G^T lies in the j -th position. For example, when NMF is applied to document clustering, the basis vectors in C represent k topics, and the coefficients in the i -th column of G^T indicate the degrees of membership for x_i , the i -th document. NMF is well-known for the *interpretability* of the latent space it finds [38].

Another way to illustrate the capability of NMF as a clustering method is by observing its relationship to the objective function of the classical K-means clustering, which is arguably the most commonly used clustering method:

$$\min \sum_{i=1}^n \|x_i - c_{g_i}\|_2^2, \tag{4}$$

where x_1, \dots, x_n are the columns of X , c_1, \dots, c_k are the k centroids, and $g_i = j$ when the i -th data point is assigned to the j -th cluster ($1 \leq j \leq k$). Consider K-means formulated as a dimension reduction problem [30]:

$$\min_{G \in \{0,1\}^{n \times k}, G\mathbf{1}_k = \mathbf{1}_n} \|X - CG^T\|_F^2, \tag{5}$$

where $\mathbf{1}_k \in \mathbb{R}^{k \times 1}$, $\mathbf{1}_n \in \mathbb{R}^{n \times 1}$ are column vectors whose elements are all 1's. In the formulation (5), columns of C are the cluster centroids, and the single nonzero element in each column of G^T indicates the clustering assignment.

NMF as a clustering method has been proved to be superior to K-means on many types of data, including documents [64], images [8], and microarray data [27]. Although K-means and NMF have the same objective function $\|X - CG^T\|_F^2$ with different constraints, i.e. $G \in \{0, 1\}^{n \times k}$, $G\mathbf{1}_k = \mathbf{1}_n$ in the case of K-means, and $C \geq 0$ and $G \geq 0$ in the case of NMF, each has its best performance on different kinds of data sets. In order to apply NMF to the appropriate data sets, we must know the limitation of its capability in clustering. Most clustering methods have a clearly defined objective function to optimize such as (5) and (3). However, clustering is difficult to formulate mathematically in order to discover the hidden pattern [33]. Each clustering method has its own conditions under which it performs well. For example, K-means assumes that data points in each cluster follow a spherical Gaussian distribution [18]. In contrast, the NMF formulation (3) provides a better low-rank approximation of the data matrix X than the K-means formulation (5).

If $k \leq \text{rank}(X)$, since $\text{rank}(X) \leq \text{nonnegative-rank}(X)$ [4] and the low rank approximation by NMF gives a smaller objective function value when the columns of C (the cluster representatives) are linearly independent, it is for the best interest of NMF to produce linearly independent cluster representatives. This explains our earlier discovery that NMF performs well when different clusters correspond to linearly independent vectors [34]. The following artificial example illustrates this point. See Fig. 1, where the two cluster centers are along the same direction therefore the two centroid vectors are linearly dependent. While NMF still approximates all the data points well in this example, no two linearly independent vectors in a two-dimensional space can represent the two clusters shown in Fig. 1. Since K-means and NMF have different conditions under which each of them does clustering well, they may generate very different clustering results in practice. We are motivated by Fig. 1 to mention that the assumption of spherical K-means is that data points in each cluster follow a von Mises–Fisher distribution [3], which is similar to the assumption of NMF.

Therefore, NMF, originally a dimension reduction method, is not always a preferred clustering method. The success of NMF as a clustering method depends on the underlying data set, and its most success has been around document clustering [17, 30, 41, 52, 55, 64]. In a document data set, data points are often represented as unit-length vectors [47] and embedded in a linear subspace. For a term-document matrix X , a basis vector c_j is interpreted as the term distribution of a single topic. As long as the term distributions of k topics are linearly independent, which are usually the case, NMF can extract the ground-truth clusters well. However, NMF has not been as successful in image clustering. For image data, it was shown

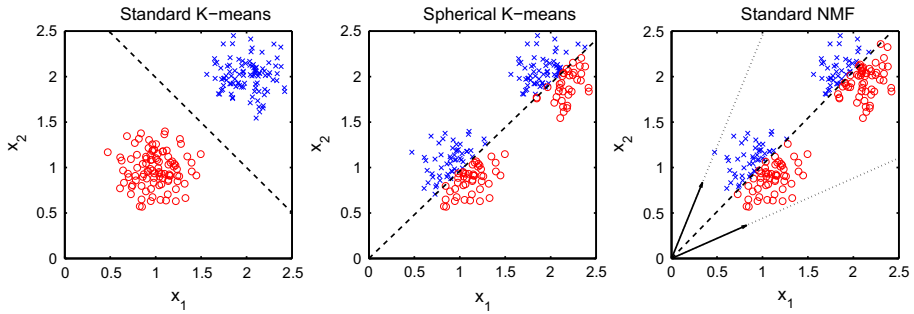


Fig. 1 An example with two ground-truth clusters, with different clustering results

that a collection of images tends to form multiple 1-dimensional nonlinear manifolds [60], one manifold for each cluster. This does not satisfy NMF’s assumption on cluster structures, and therefore NMF may not identify correct clusters.

In this paper, we study a more general formulation for clustering based on NMF, called *Symmetric NMF* (SymNMF), where an $n \times n$ nonnegative and symmetric matrix A is given as an input instead of a nonnegative data matrix X . The matrix A contains pairwise similarity values of a similarity graph, and is approximated by a lower rank matrix HH^T instead of the product of two lower rank matrices CG^T . High-dimensional data such as documents and images are often embedded in a low-dimensional space, and the embedding can be extracted from their graph representation. We will demonstrate that SymNMF can be used for graph embedding and clustering and often performs better than spectral methods in terms of multiple evaluation measures for clustering.

The rest of this paper is organized as follows. In Sect. 2, we review previous work on nonnegative factorization of a symmetric matrix and introduce the novelty of the directions proposed in this paper. In Sect. 3, we present our new interpretation of SymNMF as a clustering method. In Sect. 4, we show the difference between SymNMF and spectral clustering in terms of their dependence on the spectrum. In Sects. 5 and 6, we propose two algorithms for SymNMF: A Newton-like algorithm and an alternating nonnegative least squares (ANLS) algorithm, and discuss their efficiency and convergence properties. In Sect. 7, we report experiment results on document and image clustering that illustrate that SymNMF is a competitive method for graph clustering. In Sect. 8, we apply SymNMF to image segmentation and show the unique properties of the obtained segments. In Sect. 9, we discuss future research directions.

2 Related work

In SymNMF, we look for the solution $H \in \mathbb{R}_+^{n \times k}$,

$$\min_{H \geq 0} f(H) = \|A - HH^T\|_F^2, \tag{6}$$

given $A \in \mathbb{R}_+^{n \times n}$ with $A^T = A$ and k . The integer k is typically much smaller than n . In our graph clustering setting, A is called a *similarity matrix*: The (i, j) -th entry of A is the similarity value between the i -th and j -th nodes in a similarity graph, or the similarity value between the i -th and j -th data items.

The above formulation has been studied in a number of previous papers. Ding et al. [15] transformed the formulation of NMF (3) to a symmetric approximation $\|A - HH^T\|_F^2$ where A is a positive semi-definite matrix, and showed that it has the same form as the objective function of spectral clustering. Li et al. [41] used this formulation for semi-supervised clustering where the similarity matrix was modified with prior information. Zass and Shashua [69] converted a completely positive matrix [5] to a symmetric doubly stochastic matrix A and used the formulation (6) to find a nonnegative H for probabilistic clustering. They also gave a reason why the nonnegativity constraint on H was more important than the orthogonality constraint in spectral clustering. He et al. [23] approximated a completely positive matrix directly using the formulation (6) with parallel update algorithms. In all of the above work, A was assumed to be a positive semi-definite matrix. Other related work that imposed additional constraints on H includes [2, 65, 66].

The SymNMF formulation has also been applied to non-overlapping and overlapping community detection in real networks [45, 50, 62, 71, 72]. For example, Nepusz et al. [50] proposed a formulation similar to (6) with sum-to-one constraints to detect soft community memberships; Zhang et al. [72] proposed a binary factorization model for overlapping communities and discussed the pros and cons of hard/soft assignments to communities. The adjacency matrix A involved in community detection is often an indefinite matrix.

Catral et al. [9] studied whether WH^T is symmetric and $W = H$, when W and H are the global optimum for the problem $\min_{W, H \geq 0} \|A - WH^T\|_F^2$ where A is nonnegative and symmetric. Ho [24] in his thesis related SymNMF to the exact SymNMF problem $A = HH^T$. Both of their theories were developed outside the context of graph clustering, and their topics are beyond the scope of this paper. Ho [24] also proposed a $2n$ -block coordinate descent algorithm for (6). Compared to our two-block coordinate descent framework described in Sect. 6, Ho's approach introduced a dense $n \times n$ matrix which destroys the sparsity pattern in A and is not scalable.

Almost all the work mentioned above employed multiplicative update algorithms to optimize their objective functions with nonnegativity constraints. However, this type of algorithms does not have the property that every limit point is a stationary point [22, 42], and accordingly their solutions are not guaranteed to be local minima. In fact, the results of multiplicative update algorithms (e.g. [16]) only satisfy part of the KKT condition, but do not satisfy all the components of the KKT condition, for example, the sign of the gradient vector. In the three papers [24, 50, 71] that used gradient descent methods for optimization that *did* reach stationary point solutions, they performed the experiments only on graphs with up to thousands of nodes.

In this paper, we study the formulation (6) from different angles:

1. We focus on a more general case where A is a symmetric indefinite matrix and represents a general graph. Examples of such an indefinite matrix include a similarity matrix for high-dimensional data formed by the self-tuning method [70] as well as the pixel similarity matrix in image segmentation [56]. Real networks have additional structures such as the scale-free properties [59], and we will not include them in this work.
2. We focus on hard clustering and will give an intuitive interpretation of SymNMF as a graph clustering method. Hard clustering offers more explicit membership and easier visualization than soft clustering [72]. Unlike [15], we emphasize the difference between SymNMF and spectral clustering instead of their resemblance.
3. We propose two optimization algorithms that converge to stationary point solutions for SymNMF, namely Newton-like algorithm and ANLS algorithm. We also show that the new ANLS algorithm scales better to large data sets.

- In addition to experiments on document and image clustering, we apply SymNMF to image segmentation using 200 images in the Berkeley Segmentation Data Set [1]. To the best of our knowledge, our work represents the first attempt for a thorough evaluation of nonnegativity-based methods for image segmentation.

Overall, we conduct a comprehensive study of SymNMF in this paper, covering from foundational justification for SymNMF for clustering, convergent and scalable algorithms, to real-life applications for text and image clustering as well as image segmentation. The Newton-like algorithm and some of the analysis of spectral clustering was first proposed in our previous work [34]. We include them in this paper for completeness.

3 Interpretation of SymNMF as a graph clustering method

Just as the nonnegativity constraint in NMF makes it interpretable as a clustering method, the nonnegativity constraint $H \geq 0$ in (6) also gives a natural partitioning as well as interpretation of SymNMF. Now we provide an intuitive explanation of why this formulation is expected to extract cluster structures.

Figure 2 shows an illustrative example of SymNMF, where we have rearranged the rows and columns of A without loss of generality. If a similarity matrix has cluster structures embedded in it, several diagonal blocks (two diagonal blocks in Fig. 2) with large similarity values will appear. In order to approximate this similarity matrix with low-rank matrices and simultaneously extract cluster structures, we can approximate these diagonal blocks separately because each diagonal block indicates one cluster. As shown in Fig. 2, it is straightforward to use an outer product hh^T to approximate a diagonal block. Because h is a nonnegative vector, it serves as a cluster membership indicator: Larger values in h indicate stronger memberships to the cluster corresponding to the diagonal block. When multiple such outer products are added up together, they approximate the original similarity matrix, and each column of H represents one cluster. Due to the nonnegativity constraints in SymNMF, only “additive”, or “non-subtractive”, summation of rank-1 matrices is allowed to approximate both diagonal and off-diagonal blocks.

On the contrary, Fig. 3 illustrates the result of low-rank approximation of A without nonnegativity constraints. In this case, when using multiple hh^T outer products to approximate A , cancellations of positive and negative numbers are allowed. Without nonnegativity enforced on h 's, the diagonal blocks need not be approximated separately by each term hh^T . The elements in a vector h can have any sign (+, 0, -) and magnitude, though the summation of all hh^T terms, i.e. HH^T , can approximate the large diagonal blocks and small off-diagonal blocks well. Thus, h cannot serve as a cluster membership indicator. In this case, the rows of the low-rank matrix H contain both positive and negative numbers and can be used for graph embedding. In order to obtain hard clusters, we need to post-process the embedded

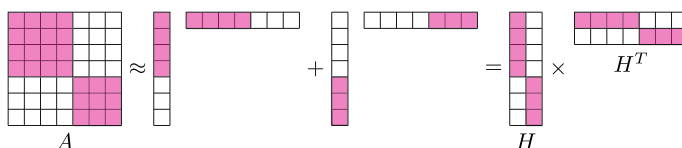


Fig. 2 An illustration of SymNMF formulation $\min_{H \geq 0} \|A - HH^T\|_F^2$. Each cell is a matrix entry. Colored region has larger values than white region. Here $n = 7$ and $k = 2$. (Color figure online)

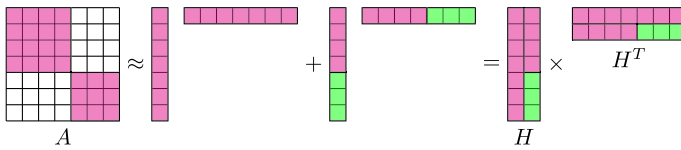


Fig. 3 An illustration of $\min \|A - HH^T\|_F^2$ or $\min_{HH^T=I} \|A - HH^T\|_F^2$. Each cell is a matrix entry. *Colored region* has larger magnitudes than *white region*. *Magenta cells* indicate positive entries, *green* indicating negative. Here $n = 7$ and $k = 2$. (Color figure online)

data points such as applying K-means clustering. This reasoning is analogous to the contrast between NMF and SVD [38].

SymNMF is flexible in terms of choosing similarities between data points. We can choose any similarity measure that describes the cluster structure well. In fact, the formulation of NMF (3) can be related to SymNMF when $A = X^T X$ in (6) [15]. This means that NMF implicitly chooses inner products as the similarity measure, which is not always suitable to distinguish different clusters.

4 SymNMF and spectral clustering

4.1 Objective functions

Spectral clustering represents a large class of graph clustering methods that rely on eigenvector computation [10,51,56]. Now we will show that spectral clustering and SymNMF are closely related in terms of the graph clustering objective but fundamentally different in optimizing this objective.

Many graph clustering objectives can be reduced to a trace maximization form [13,36]:

$$\max \text{trace}(\tilde{H}^T A \tilde{H}), \tag{7}$$

where $\tilde{H} \in \mathbb{R}^{n \times k}$ (to be distinguished from H in the SymNMF formulation) satisfies $\tilde{H}^T \tilde{H} = I$, $\tilde{H} \geq 0$, and each row of \tilde{H} contains one positive entry and at most one positive entry due to $\tilde{H}^T \tilde{H} = I$. Clustering assignments can be drawn from \tilde{H} accordingly.

Under the constraints on $\tilde{H}^T \tilde{H} = I$, $\tilde{H} \geq 0$, we have [15]:

$$\begin{aligned} &\max \text{trace}(\tilde{H}^T A \tilde{H}) \\ &\Leftrightarrow \min \text{trace}(A^T A) - 2\text{trace}(\tilde{H}^T A \tilde{H}) + \text{trace}(I) \\ &\Leftrightarrow \min \text{trace} \left[(A - \tilde{H} \tilde{H}^T)^T (A - \tilde{H} \tilde{H}^T) \right] \\ &\Leftrightarrow \min \|A - \tilde{H} \tilde{H}^T\|_F^2. \end{aligned}$$

This objective function is the same as (6), except that the constraints on the low-rank matrices H and \tilde{H} are different. The constraint on \tilde{H} makes the graph clustering problem NP-hard [56], therefore a practical method relaxes the constraint to obtain a tractable formulation. In this respect, spectral clustering and SymNMF can be seen as two different ways of relaxation: While spectral clustering retains the constraint $\tilde{H}^T \tilde{H} = I$, SymNMF retains $\tilde{H} \geq 0$ instead. These two choices lead to different algorithms for optimizing the same graph clustering objective (7), which are shown in Table 1.

Table 1 Algorithmic steps of spectral clustering and SymNMF clustering

	Spectral clustering	SymNMF
Objective	$\min_{\hat{H}} \ \hat{H} - \hat{H} \hat{H}^T\ _F^2$	$\min_{H \geq 0} \ A - HH^T\ _F^2$
Step 1	Obtain the global optimal $\hat{H}_{n \times k}$ by computing k leading eigenvectors of A	Obtain a solution H using an optimization algorithm
Step 2	Scale each row of \hat{H}	(No need to scale rows of H)
Step 3	Apply a clustering algorithm to the rows of \hat{H} , a k -dimensional embedding	The largest entry in each row of H indicates the clustering assignments

4.2 Spectral clustering and the spectrum

Normalized cut is a widely-used objective for spectral clustering [56]. Now we describe some scenarios where optimizing this objective may have difficulty in identifying correct clusters while SymNMF could be potentially better.

Although spectral clustering is a well-established framework for graph clustering, its success relies only on the properties of the leading eigenvalues and eigenvectors of the similarity matrix A . It was pointed out in [51, 58] that the k -dimensional subspace spanned by the leading k eigenvectors of A is stable only when $|\lambda_k(A) - \lambda_{k+1}(A)|$ is sufficiently large, where $\lambda_i(A)$ is the i -th largest eigenvalue of A . Now we show that spectral clustering could fail when this condition is not satisfied but the cluster structure may be perfectly represented in the block-diagonal structure of A . Suppose A is composed of $k = 3$ diagonal blocks, corresponding to three clusters:

$$A = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{bmatrix}. \tag{8}$$

If we construct A as in the normalized cut, then each of the diagonal blocks A_1, A_2, A_3 has a leading eigenvalue 1. We further assume that $\lambda_2(A_i) < 1$ for all $i = 1, 2, 3$ in exact arithmetic. Thus, the three leading eigenvectors of A correspond to the diagonal blocks A_1, A_2, A_3 respectively. However, when $\lambda_2(A_1)$ and $\lambda_3(A_1)$ are so close to 1 that it cannot be distinguished from $\lambda_1(A_1)$ in finite precision arithmetic, it is possible that the computed eigenvalues $\tilde{\lambda}_j(A_i)$ satisfy $\tilde{\lambda}_1(A_1) > \tilde{\lambda}_2(A_1) > \tilde{\lambda}_3(A_1) > \max(\tilde{\lambda}_1(A_2), \tilde{\lambda}_1(A_3))$. In this case, three subgroups are identified within the first cluster; the second and the third clusters cannot be identified, as shown in Fig. 4 where all the data points in the second and third clusters are mapped to $(0, 0, 0)$. Therefore, eigenvectors computed in a finite precision cannot always capture the correct low-dimensional graph embedding.

Now we demonstrate the above scenario using a concrete graph clustering example. Figure 5 shows (a) the original data points; (b) the embedding generated by spectral clustering; and (c, d) plots of the similarity matrix A . Suppose the scattered points form the first cluster, and the two tightly-clustered groups correspond to the second and third clusters. We employ the widely-used Gaussian kernel [61] and normalized similarity values [56]:

Fig. 4 Three leading eigenvectors of the similarity matrix in (8) when $\tilde{\lambda}_3(A_1) > \max(\tilde{\lambda}_1(A_2), \tilde{\lambda}_1(A_3))$. Here we assume that all the block diagonal matrices A_1, A_2, A_3 have size 3×3 . Colored region has nonzero values. (Color figure online)

0	0	0
0	0	0
0	0	0
0	0	0
0	0	0
0	0	0

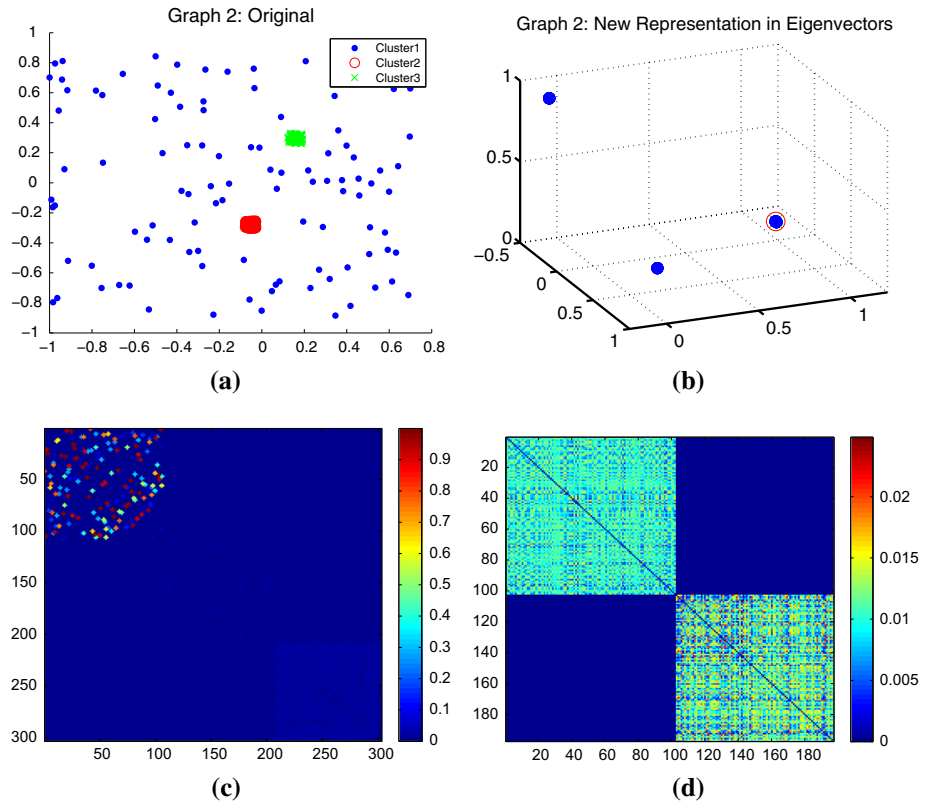


Fig. 5 A graph clustering example with three clusters (original data from [70]). **a** Data points in the original space. For illustration’s purpose, we use two-dimensional data points; however, the Gaussian kernel (9) maps them into a high-dimensional space. **b** 3-dimensional embedding of the data points as rows of three leading eigenvectors. **c** Block-diagonal structure of A . **d** Block-diagonal structure of the submatrix of A corresponding to the two tightly-clustered groups in (a). Note that the data points in both **a** and **b** are marked with ground-truth labels

$$e_{ij} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{\sigma^2}\right),$$

$$A_{ij} = e_{ij}d_i^{-1/2}d_j^{-1/2}, \tag{9}$$

where x_i ’s are the two-dimensional data points, $d_i = \sum_{s=1}^n e_{is}(1 \leq i \leq n)$, and σ is a parameter set to 0.05 based on the scale of data points. In spectral clustering, the rows of

Table 2 Leading eigenvalues of the similarity matrix based on Fig. 5 with $\sigma = 0.05$

1st	1.000000000000001
2nd	1.000000000000000
3rd	1.000000000000000
4th	0.9999999999998909

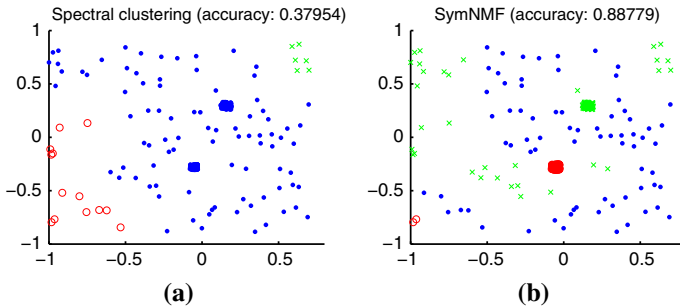


Fig. 6 Clustering results for the example in Fig. 5: **a** Spectral clustering, **b** SymNMF

the leading eigenvectors determine a mapping of the original data points, shown in Fig. 5b. In this example, the original data points are mapped to three unique points in a new space. However, the three points in the new space do not correspond to the three clusters in Fig. 5a. In fact, out of the 303 data points in total, 290 data points are mapped to a single point in the new space.

Let us examine the leading eigenvalues, shown in Table 2, where the fourth largest eigenvalue of A is very close to the third largest eigenvalue. This means that the second largest eigenvalue of a cluster, say $\lambda_2(A_1)$, would be easily identified as one of $\lambda_1(A_1)$, $\lambda_1(A_2)$, and $\lambda_1(A_3)$. The mapping of the original data points shown in Fig. 5b implies that the computed three largest eigenvalues come from the first cluster. This example is a noisier case of the scenario in Fig. 4.

On the contrary, we can see from Fig. 5c, d that the block-diagonal structure of A is clear, though the within-cluster similarity values are not on the same scale. Figure 6 shows the comparison of clustering results of spectral clustering and SymNMF in this case. SymNMF is able to separate the two tightly-clustered groups more accurately.

4.3 A condition on SymNMF

How does the spectrum of A possibly influence SymNMF? We have seen that the solution of SymNMF relies on the block-diagonal structure of A , thus it does not suffer from the situations in Sect. 4.2 where the eigengap between the k -th and $(k + 1)$ -th eigenvalues is small. We will also see in later sections that algorithms for SymNMF do not depend on eigenvector computation. However, we do emphasize a condition that SymNMF must satisfy in order to make the formulation (6) valid. This condition is related to the spectrum of A , specifically the number of nonnegative eigenvalues of A . Note that A is only assumed to be symmetric and nonnegative, and is not necessarily positive semi-definite, therefore may have both positive and negative eigenvalues. On the other hand, in the approximation $\|A - HH^T\|_F$, HH^T is always positive semi-definite and has rank at most k , therefore HH^T would not be a good

Algorithm 1 Framework of the Newton-like algorithm for SymNMF: $\min_{H \geq 0} f(x) = \|A - HH^T\|_F^2$

- 1: Input: number of data points n , number of clusters k , $n \times n$ similarity matrix A , reduction factor $0 < \beta < 1$, acceptance parameter $0 < \sigma < 1$, and tolerance parameter $0 < \mu \ll 1$
- 2: Initialize $x, x^{(0)} \leftarrow x$
- 3: **repeat**
- 4: **Compute scaling matrix** S
- 5: Step size $\alpha = 1$
- 6: **while** true **do**
- 7: $x_{\text{new}} = [x - \alpha S \nabla f(x)]^+$
- 8: **if** $f(x_{\text{new}}) - f(x) \leq \sigma \nabla f(x)^T (x_{\text{new}} - x)$ **then**
- 9: break
- 10: **end if**
- 11: $\alpha \leftarrow \beta \alpha$
- 12: **end while**
- 13: $x \leftarrow x_{\text{new}}$
- 14: **until** $\|\nabla^P f(x)\| \leq \mu \|\nabla^P f(x^{(0)})\|$
- 15: Output: x

approximation if A has fewer than k nonnegative eigenvalues. We assume that A has at least k nonnegative eigenvalues when the given size of H is $n \times k$.

This condition on A could be expensive to check. Here, by a simple argument, we claim that it is practically reasonable to assume that this condition is satisfied given a similarity matrix. Again, we use the similarity matrix A in (8) as an example. Suppose we know the actual number of clusters is three, and therefore H has size $n \times 3$. Because A is nonnegative, each of A_1, A_2, A_3 has at least one nonnegative eigenvalue according to Perron–Frobenius theorem [4], and A has at least three nonnegative eigenvalues. In a real data set, A may become much noisier with small entries in the off-diagonal blocks of A . The eigenvalues are not dramatically changed by a small perturbation of A according to matrix perturbation theory [58], hence A would also have at least k nonnegative eigenvalues if its noiseless version does. In practice, the number of positive eigenvalues of A is usually much larger than that of negative eigenvalues, which is verified in our experiments.

5 A Newton-like algorithm for SymNMF

In this section, we will present an optimization algorithm to compute SymNMF where A is nonnegative and symmetric. The objective function in (6) is a fourth-order non-convex function with respect to the entries of H , and has multiple local minima. For this type of problem, it is difficult to find a global minimum; thus a good convergence property we can expect is that every limit point is a stationary point [7]. We could directly apply standard gradient search algorithms, which lead to stationary point solutions; however, they suffer from either slow convergence or expensive computation cost.

5.1 Algorithm framework

First, we introduce our notations for clarity. Let $H = [h_1, \dots, h_k] \in \mathbb{R}_+^{n \times k}$. A vector x of length nk is used to represent the vectorization of H by column, i.e. $x = \text{vec}(H) = [h_1^T, \dots, h_k^T]^T \in \mathbb{R}_+^{nk \times 1}$. For simplicity, functions applied on x have the same notation as functions applied on H , i.e. $f(x) \equiv f(H)$. $[\cdot]^+$ denotes the projection to the nonnegative

Table 3 Comparison of PGD and PNewton for solving $\min_{H \geq 0} \|A - HH^T\|_F^2$, $H \in \mathbb{R}_+^{n \times k}$

	Projected gradient descent (PGD)	Projected Newton (PNewton)
Scaling matrix	$S^{(t)} = I_{nk \times nk}$	$S^{(t)} = \left(\nabla_{\mathcal{E}}^2 f(x^{(t)})\right)^{-1}$
Convergence	Linear (zigzagging)	Quadratic
Complexity	$O(n^2k)$ /iteration	$O(n^3k^3)$ /iteration

orthant, i.e. replacing any negative element of a vector to be 0. Superscripts denote iteration indices, e.g. $x^{(t)} = \text{vec}(H^{(t)})$ is the iterate of x in the t -th iteration. For a vector v , v_i denotes its i -th element. For a matrix M , M_{ij} denotes its (i, j) -th entry; and $M_{[i][j]}$ denotes its (i, j) -th $n \times n$ block, assuming that both the numbers of rows and columns of M are multiples of n . $M \succ 0$ refers to positive definiteness of M . We define the *projected gradient* $\nabla^P f(x)$ at x as [43]:

$$\left(\nabla^P f(x)\right)_i = \begin{cases} (\nabla f(x))_i, & \text{if } x_i > 0; \\ [(\nabla f(x))_i]^+, & \text{if } x_i = 0. \end{cases} \tag{10}$$

Algorithm 1 describes a framework of gradient search algorithms applied to SymNMF, based on which we developed our Newton-like algorithm. This description does not specify iteration indices, but updates x in-place. The framework uses the “scaled” negative gradient direction as search direction. Except the scalar parameters β, σ, μ , the $nk \times nk$ scaling matrix $S^{(t)}$ is the only unspecified quantity. Table 3 lists two choices of $S^{(t)}$ that lead to different gradient search algorithms: projected gradient descent (PGD) [43] and projected Newton (PNewton) [7].

PGD sets $S^{(t)} = I$ throughout all the iterations. It is known as one of steepest descent methods, and does not scale the gradient using any second-order information. This strategy often suffers from the well-known zigzagging behavior, thus has slow convergence rate [7]. On the other hand, PNewton exploits second-order information provided by the Hessian $\nabla^2 f(x^{(t)})$ as much as possible. PNewton sets $S^{(t)}$ to be the inverse of a reduced Hessian at $x^{(t)}$. The reduced Hessian with respect to index set R is defined as:

$$\left(\nabla_R^2 f(x)\right)_{ij} = \begin{cases} \delta_{ij}, & \text{if } i \in R \text{ or } j \in R; \\ \left(\nabla^2 f(x)\right)_{ij}, & \text{otherwise,} \end{cases} \tag{11}$$

where δ_{ij} is the Kronecker delta. Both the gradient and the Hessian of $f(x)$ can be computed analytically:

$$\begin{aligned} \nabla f(x) &= \text{vec}\left(4(HH^T - A)H\right), \\ \left(\nabla^2 f(x)\right)_{[i][j]} &= 4\left(\delta_{ij}(HH^T - A) + h_j h_i^T + (h_i^T h_j)I_{n \times n}\right). \end{aligned}$$

We introduce the definition of an index set \mathcal{E} that helps to prove the convergence of Algorithm 1 [7]:

$$\mathcal{E} = \{i | 0 \leq x_i \leq \epsilon, (\nabla f(x))_i > 0\}, \tag{12}$$

where ϵ depends on x and is usually small ($0 < \epsilon < 0.01$) [26]. In PNewton, $S^{(t)}$ is formed based on the reduced Hessian $\nabla_{\mathcal{E}}^2 f(x^{(t)})$ with respect to \mathcal{E} . However, because the computation of the scaled gradient $S^{(t)}\nabla f(x^{(t)})$ involves the Cholesky factorization of the reduced Hessian, PNewton has a very large computational complexity of $O(n^3k^3)$, which

is prohibitive. Therefore, we propose a Newton-like algorithm that exploits second-order information in an inexpensive way.

5.2 Improving the scaling matrix

The choice of the scaling matrix $S^{(t)}$ is essential to an algorithm that can be derived from the framework described in Algorithm 1. We propose two improvements on the choice of $S^{(t)}$, yielding new algorithms for SymNMF. Our focus is to efficiently collect partial second-order information but meanwhile still effectively guide the scaling of the gradient direction. Thus, these improvements seek a tradeoff between convergence rate and computational complexity, with the goal of accelerating SymNMF algorithms as an overall outcome.

Our design of new algorithms must guarantee the convergence. Since the algorithm framework still follows Algorithm 1, we would like to know what property of the scaling matrix $S^{(t)}$ is essential in the proof of the convergence result of PGD and PNewton. This property is described by the following lemma:

Definition 1 A scaling matrix S is diagonal with respect to an index set R , if $S_{ij} = 0, \forall i \in R$ and $j \neq i$ [6].

Lemma 1 Let S be a positive definite matrix which is diagonal with respect to \mathcal{E} . If $x \geq 0$ is not a stationary point, there exists $\bar{\alpha} > 0$ such that $f([x - \alpha S \nabla f(x)]^+) < f(x), \forall 0 < \alpha < \bar{\alpha}$. (modified from [6])

Lemma 1 states the requirement on $S^{(t)}$, which is satisfied by the choices of $S^{(t)}$ in both PGD and PNewton. It guides our development of new ways to choose $S^{(t)}$.

5.2.1 Improvement 1: fewer Hessian evaluations

A common method for reducing computation cost related to $S^{(t)}$ is to periodically update $S^{(t)}$ or evaluate $S^{(t)}$ only at the 1st iteration (chord method) [26]. However, this method cannot be directly used in the framework of Algorithm 1, because $S^{(t)}$ is not necessarily diagonal with respect to $\mathcal{E}^{(t)}$ if $\mathcal{E}^{(t)} \neq \mathcal{E}^{(t-1)}$, and the requirement for convergence is violated.

Our way to delay the update of $S^{(t)}$ is to evaluate $S^{(t)}$ only when $\mathcal{E}^{(t)}$ changes. More precisely,

$$S^{(t)} = \begin{cases} S^{(t-1)}, & \text{if } \mathcal{E}^{(t)} = \mathcal{E}^{(t-1)}; \\ (\nabla_{\mathcal{E}}^2 f(x^{(t)}))^{-1}, & \text{if } \mathcal{E}^{(t)} \neq \mathcal{E}^{(t-1)} \\ & \text{and } \nabla_{\mathcal{E}}^2 f(x^{(t)}) \succ 0; \\ I_{nk \times nk}, & \text{otherwise.} \end{cases} \tag{13}$$

Note that because $f(x)$ is non-convex, we have to set $S^{(t)} = I$ when $\nabla_{\mathcal{E}}^2 f(x^{(t)})$ is not positive definite, which can be checked during its Cholesky factorization. We expect that this improvement can reduce the number of Hessian evaluations and Cholesky factorizations.

5.2.2 Improvement 2: cheaper Hessian evaluations

The second improvement in choosing $S^{(t)}$ is inspired by the recently proposed *coordinate gradient descent* (CGD) method for solving covariance selection [68]. When CGD is directly applied to SymNMF, it updates one column of H in each iteration while the other columns are fixed, and the search direction is typically determined by solving a quadratic programming problem. The CGD method introduces additional overhead when determining the search

direction; however, it implies a possibility of using second-order information without evaluating the entire Hessian.

Inspired by the incremental update framework of CGD, we propose to choose $S^{(t)}$ to be a block-diagonal matrix in our batch update framework in Algorithm 1. Specifically,

$$S_{[i][j]}^{(t)} = \begin{cases} 0, & \text{if } i \neq j; \\ (\nabla_{\xi}^2 f(x^{(t)})_{[i][j]})^{-1}, & \text{if } i = j \\ & \text{and } \nabla_{\xi}^2 f(x^{(t)})_{[i][j]} > 0; \\ I_{n \times n}, & \text{otherwise.} \end{cases} \quad (14)$$

Intuitively speaking, the i -th $n \times n$ diagonal block of $S^{(t)}$ corresponds to variables in the i -th column of H , and $S^{(t)}$ only involves second-order information within each column of H . This choice of $S^{(t)}$ has two advantages over the choice in PNewton algorithm: First, the computational complexity in each iteration is $O(n^3k)$, much lower than the complexity of PNewton if k is not too small. Second, we can exploit partial second-order information even though the n diagonal blocks of $\nabla_{\xi}^2 f(x^{(t)})$ are not all positive definite, whereas PNewton requires the positive definiteness of all the n diagonal blocks as a necessary condition.

Our final strategy for solving SymNMF (6) is to combine Improvement 1 and Improvement 2. Note that the requirement on $S^{(t)}$ described in Lemma 1 is satisfied in both of the improvements, and also in their combination. Thus, convergence is guaranteed in all of these variations.

6 An ANLS algorithm for SymNMF

In this section, we propose another optimization algorithm for SymNMF that converges to stationary points, a necessary condition for local minima. The algorithm is based on an alternative formulation of SymNMF, where it becomes straightforward to use the two-block coordinate descent framework that has been shown efficient for standard NMF [28,29,31,32,43].

6.1 Two-block coordinate descent framework

We first briefly review the *two-block coordinate descent framework* [28,29,32,43] for standard NMF problems shown in (3):

$$\min_{C \geq 0, G \geq 0} \|X - CG^T\|_F^2,$$

which has our desired convergence property that every limit point is a stationary point. Separating the unknowns in C and G in the NMF formulation (3) into two blocks, we obtain the following subproblems:

1. Fix G and solve $\min_{C \geq 0} \|GC^T - X^T\|_F^2$.
2. Fix C and solve $\min_{G \geq 0} \|CG^T - X\|_F^2$.

Each subproblem is a nonnegative least squares problem with multiple right-hand sides (NLS for short), and many efficient procedures have been developed to solve NLS, e.g. active-set method [28,37], block pivoting [31,32], PGD [43], etc. The key requirement in this framework is to obtain the optimal solution in each subproblem (see more discussions in [28]). This way, the original NMF formulation (3) has been reduced to an alternating NLS problem (ANLS for short).

Algorithm 2 Framework of the ANLS algorithm for SymNMF: $\min_{W, H \geq 0} \|A - WH^T\|_F^2 + \alpha \|W - H\|_F^2$

- 1: Input: number of data points n , number of clusters k , $n \times n$ similarity matrix A , regularization parameter $\alpha > 0$, and tolerance parameter $0 < \mu < 1$
- 2: Initialize H
- 3: **repeat**
- 4: $W \leftarrow H$
- 5: **Solve an NLS problem:** $H \leftarrow \arg \min_{H \geq 0} \left\| \begin{bmatrix} W \\ \sqrt{\alpha} I_k \end{bmatrix} H^T - \begin{bmatrix} A \\ \sqrt{\alpha} W^T \end{bmatrix} \right\|_F$
- 6: **until** $\|\nabla^P g(W, H)\|_F \leq \mu \|\nabla^P g(W^{(0)}, H^{(0)})\|_F$
- 7: Output: H

6.2 A nonsymmetric formulation for SymNMF

In SymNMF, it is difficult to separate the nk unknowns in a straightforward way as in NMF, because the two factors H and H^T contain the same set of unknowns. We propose to reformulate SymNMF in the context of NMF [24]:

$$\min_{W, H \geq 0} g(W, H) = \|A - WH^T\|_F^2 + \alpha \|W - H\|_F^2, \tag{15}$$

where A still represents the $n \times n$ similarity matrix, W, H are two low-rank factors of size $n \times k$, and $\alpha > 0$ is a scalar parameter for the tradeoff between the approximation error and the difference of W and H . Here we *force* the separation of unknowns by associating the two factors with two different matrices. If α has a large enough value, the solutions of W and H will be close enough so that the clustering results will not be affected whether W or H are used as the clustering assignment matrix.

The nonsymmetric formulation can be easily cast into the two-block coordinate descent framework after some restructuring. In particular, we have the following subproblems for (15):

$$\min_{W \geq 0} \left\| \begin{bmatrix} H \\ \sqrt{\alpha} I_k \end{bmatrix} W^T - \begin{bmatrix} A \\ \sqrt{\alpha} H^T \end{bmatrix} \right\|_F, \tag{16}$$

$$\min_{H \geq 0} \left\| \begin{bmatrix} W \\ \sqrt{\alpha} I_k \end{bmatrix} H^T - \begin{bmatrix} A \\ \sqrt{\alpha} W^T \end{bmatrix} \right\|_F, \tag{17}$$

where $\mathbf{1}_k \in \mathbb{R}^{k \times 1}$ is a column vector whose elements are all 1’s, and I_k is the $k \times k$ identity matrix. Note that we have assumed $A = A^T$. Solving subproblems (16) and (17) in an alternate fashion will lead to a stationary point solution, as long as an optimal solution is returned for every NLS subproblem encountered. We simplify and summarize this algorithm in Algorithm 2.

If W and H are expected to indicate more distinct cluster structures, sparsity constraints on the rows of W and H can also be incorporated into the nonsymmetric formulation easily, by adding L_1 regularization terms [27,28]:

$$\min_{W, H \geq 0} \tilde{g}(W, H) = \|A - WH^T\|_F^2 + \alpha \|W - H\|_F^2 + \beta \sum_{i=1}^n \|w_i\|_1^2 + \beta \sum_{i=1}^n \|h_i\|_1^2, \tag{18}$$

where $\alpha, \beta > 0$ are regularization parameters, w_i, h_i are the i -th rows of W, H respectively, and $\|\cdot\|_1$ denotes vector 1-norm. Consequently, the two subproblems for (18) in the two-block

coordinate descent framework are:

$$\min_{W \geq 0} \left\| \begin{bmatrix} H \\ \sqrt{\alpha} I_k \\ \sqrt{\beta} \mathbf{1}_k^T \end{bmatrix} W^T - \begin{bmatrix} A \\ \sqrt{\alpha} H^T \\ 0 \end{bmatrix} \right\|_F, \tag{19}$$

$$\min_{H \geq 0} \left\| \begin{bmatrix} W \\ \sqrt{\alpha} I_k \\ \sqrt{\beta} \mathbf{1}_k^T \end{bmatrix} H^T - \begin{bmatrix} A \\ \sqrt{\alpha} W^T \\ 0 \end{bmatrix} \right\|_F. \tag{20}$$

We can even use just one L_1 regularization term in (18), that is, $\beta \sum_{i=1}^n \|w_i\|_1^2$ or $\beta \sum_{i=1}^n \|h_i\|_1^2$, since W and H are sufficiently close; however, using one or two L_1 regularization terms does not make much difference computationally.

6.3 Implementation

Now we describe an efficient implementation of the ANLS algorithm for SymNMF. Our algorithm reduces to solving the NLS problem in line 5 of Algorithm 2. Consider a form of NLS with simplified notation: $\min_{G \geq 0} \|CG^T - X\|_F^2$. In many algorithms for NLS, the majority of time cost comes from the computation of $C^T C$ and $X^T C$. For example, in the active-set method [28] and block pivoting method [31,32], we need to form the normal equation:

$$C^T C G^T = C^T X.$$

In PGD [43], we need to compute the gradient:

$$\nabla_G = 2G(C^T C) - 2X^T C.$$

For more details of these algorithms for NLS, please refer to the original papers [28,31,32,43]. Our strategy to solve the NLS problem in Algorithm 2 is to precompute $C^T C$ and $X^T C$:

$$C^T C = W^T W + \alpha I_k, \quad X^T C = A^T W + \alpha W$$

without forming $X = \begin{bmatrix} A \\ \sqrt{\alpha} W^T \end{bmatrix}$ directly. Though this change sounds trivial, it is very costly to form X directly when A is large and sparse, especially when A is stored in the ‘‘compressed sparse column’’ format such as in Matlab and the Python `scipy` package. In our experiments, we observed that our strategy had considerable time savings in the iterative Algorithm 2.

For choosing the parameter α , we can gradually increase α from 1 to a very large number, for example, by setting $\alpha \leftarrow 1.01\alpha$. We can stop increasing α when $\|W - H\|_F / \|H\|_F$ is negligible (say, $< 10^{-8}$).

Theoretically, both the Newton-like algorithm and the ANLS algorithm are valid algorithms for SymNMF for any nonnegative and symmetric matrix A . In practice, however, when a similarity matrix A is very sparse and the efficiencies of these two algorithms become very different. The Newton-like algorithm does not take into account the structure of SymNMF formulation (6), and a sparse input matrix A cannot contribute to speeding up the algorithm because of the formation of the dense matrix HH^T in intermediate steps. On the contrary, in the ANLS algorithm, many algorithms for the NLS subproblem [28,31,32,43] can often benefit from the sparsity of similarity matrix A automatically. This benefit comes from sparse-dense matrix multiplication inside these algorithms such as AH as well as the absence of large dense matrices such as HH^T . Therefore, we recommend using the ANLS algorithm for a sparse input matrix A .

7 Experiments on document and image clustering

In this section, we show the performances of SymNMF on a number of text and image data sets, and compare SymNMF with the standard forms and variations of NMF, spectral clustering, and K-means. The SymNMF formulation is a nonconvex minimization problem. If we apply Newton-like algorithm or ANLS algorithm which is described in Sects. 5 and 6 to SymNMF, then it can find a local minimal solution but may not find a global one. Hence we need a global optimization method. Our proposed global optimization method for experiments on document and image clustering is based on a multistart global optimization algorithm [44, 53, 54] that combines random sampling with a local search procedure. That is, we choose 20 initial points uniformly within the nonnegative orthant and a local search procedure is applied to every initial point for improving it. We use Newton-like algorithm and ANLS algorithm for our local search method. Throughout the experiments, we use Matlab 7.9 (R2009b) with an Intel Xeon X5550 quad-core processor and 24GB memory.

7.1 Data preparation

We construct a sparse graph for each data set. Using sparse graphs makes large-scale clustering possible in terms of efficiency. We take the following three steps to form the similarity matrix:

1. *Construct a complete graph.* The edge weights between graph nodes are defined according to the type of data set.
 - For text data, all the document vectors are normalized to have unit 2-norm. The edge weight is the cosine similarity between two document vectors:

$$e_{ij} = x_i^T x_j, \quad (i \neq j). \tag{21}$$

- For image data, the self-tuning method [70] is used:

$$e_{ij} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{\sigma_i \sigma_j}\right), \quad (i \neq j), \tag{22}$$

where each data point has a *local scale* σ_i , as opposed to a global scale σ in (9). σ_i is set to be the Euclidean distance between x_i and its \hat{k} -th neighbor. We use $\hat{k} = 7$ as suggested in [70].

Note that we enforce self-edge weights $e_{ii} = 0$ ($1 \leq i \leq n$) in all cases [51].

2. *Sparsify the graph.* We only keep the edges that connect a node to its q nearest neighbors. More precisely, let

$$N(i) = \{j | x_j \text{ is one of the } q \text{ nearest neighbors of } x_i, j \neq i\}. \tag{23}$$

Edge weights in the sparse graph are defined as:

$$\hat{e}_{ij} = \begin{cases} e_{ij}, & \text{if } i \in N(j) \text{ or } j \in N(i); \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

We choose $q = \lfloor \log_2 n \rfloor + 1$ as suggested in [61].

3. *Form the similarity matrix.* We compute the normalized similarity values as in the normalized cut [51]:

$$A_{ij} = \hat{e}_{ij} d_i^{-1/2} d_j^{-1/2}, \tag{25}$$

where $d_i = \sum_{s=1}^n \hat{e}_{is}$ ($1 \leq i \leq n$).

Note that the similarity matrix A constructed as above is symmetric, nonnegative, and usually indefinite.

7.2 Data sets

Document clustering was conducted on the following labeled corpuses: 1. **TDT2**¹ contains 10,212 news articles from various sources (e.g. NYT, CNN, VOA) in 1998. 2. **Reuters**² contains 21,578 news articles from the Reuters newswire in 1987. 3. From the newly-released Reuters news collection **RCV1**³ [40] that contains over 800,000 articles in 1996–1997, we selected the training set containing 23,149 articles. Labels are assigned according to a topic hierarchy, and we only considered leaf topics as valid labels. 4. The research paper collection **NIPS14-16**⁴ contains 420 NIPS papers in 2001–2003 [21], which are associated with labels indicating the technical area (algorithms, learning theory, vision science, etc). For all these data sets, documents with multiple labels are discarded in our experiments. In addition, clusters representing different topics are highly unbalanced in size. We selected the largest 20, 20, 40, 9 clusters from these data sets respectively. While TDT2 and the two Reuters data sets were well maintained, the NIPS data set was extracted from PS and PDF files, resulting in very noisy texts, which can be seen from the list of terms available online (see footnote 4). For example, its vocabulary includes many symbols frequently used in formulas which are not semantically meaningful.

Image clustering was conducted on object and face recognition data sets: 1. **COIL-20**⁵ contains gray-scale images of 20 objects, rescaled to 64×64 size. The viewpoints are equally spaced in the entire 360° range, resulting in 72 images for each object. 2. **ORL**⁶ contains 400 face images of 40 persons with different facial expressions and slightly-varying pose. 3. From **Extended YaleB**⁷ face data set (with the original YaleB data included) [39], we selected 2,414 frontal face images of 38 persons, with different illumination conditions. 4. From **PIE**⁸ face data set [57], we selected 232 frontal face images of 68 persons, with different facial expressions. Compared to other variations in PIE data set such as illumination and lighting conditions, different facial expressions represent more variations in faces and the images are embedded in multiple manifolds [60]; moreover, only 3–4 images are available for each person, which makes clustering more challenging. Though ORL and the selected subset of PIE are not large-scale, they share the same characteristics: High variations within each class, with a handful of images per class. For all the image data sets, the identity information of the objects or faces is used as ground-truth labels. The statistics of the processed document and image data sets are summarized in Table 4.

7.3 Algorithms for comparison

We experimented with a large variety of clustering algorithms for a comprehensive comparison. The algorithms in our experiment can be divided into four categories:

¹ <http://projects.ldc.upenn.edu/TDT2/>.

² <http://www.daviddlewis.com/resources/testcollections/reuters21578/>.

³ http://jmlr.csail.mit.edu/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm.

⁴ <http://robotics.stanford.edu/~gal/data.html>.

⁵ <http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php>.

⁶ <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.

⁷ <http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html>.

⁸ http://www.ri.cmu.edu/research_project_detail.html?project_id=418&menu_id=261.

Table 4 Data sets used in experiments

Data set	Dimension	# Data points	# Clusters
TDT2	26,618	8,741	20
Reuters	12,998	8,095	20
RCV1	20,338	15,168	40
NIPS14-16	17,583	420	9
COIL-20	64 × 64	1,440	20
ORL	69 × 84	400	40
Extended YaleB	56 × 64	2,414	38
PIE-expression	64 × 64	232	68

1. *K-means variants* (All these K-means variants include a *batch-update* phase and an additional *online-update* phase in each run [18]. We use both phases.)

- *Standard K-means (KM)* The input matrix is constructed as follows. For text data, each column of the tf-idf matrix X [47] is scaled to have unit 2-norm; in addition, X is transformed into its normalized-cut weighted version $XD^{-1/2}$ [64], where D is defined in Sect. 3 with $e_{ij} = x_i^T x_j$. For image data, each column of X is scaled to the $[0, 1]$ interval.
- *Spherical K-means (SKM)* Unlike standard K-means that uses Euclidean distance as the dissimilarity measure, spherical K-means uses $1 - \cos(x_i, x_j)$; therefore any scaling of columns of X does not take effect. Spherical K-means was proposed for document clustering, where cosine similarity is often a better measure than Euclidean distance [14]. As mentioned in Sect. 1, we believe that spherical K-means has a closer relationship to NMF than standard K-means.
- *Kernel K-means (KKM)* Kernel K-means is a graph clustering method based on K-means. We use the weighted kernel K-means algorithm described in [13,36] that minimizes the normalized cut objective. Because K is generally indefinite, the condition for convergence is violated. We terminate the algorithm as soon as the objective function value stops decreasing.

2. *NMF variants*

- *NMF* We use the ANLS algorithm with block pivoting method for NMF [31,32]. The same input matrix as in standard K-means is used. The hard clustering result is indicated by the largest entry in each row of H .
- *GNMF* Cai et al. [8] proposed Graph-regularized NMF (GNMF) by adding a graph-theoretic penalty term to (3) that takes neighboring relationship into account, so that the resulting method is better at clustering on manifolds. We use the algorithm and the parameters suggested in [8]. The input matrix is constructed in the same way as in standard K-means. However, the neighboring relationship based on the sparse graph is generated using the original data matrix, i.e. without the scaling of each x_i . The clustering result is obtained by treating the rows of H as graph embedding and applying spherical K-means to the embedded points.

3. *Spectral clustering variants*

- *NJW algorithm (SpNJW)* This refers to the algorithm proposed in Ng et al. [51]. The rows of the k leading eigenvectors of A , where each row is normalized to have unit

2-norm, are used as the graph embedding of data points. Standard K-means is used in the final step to obtain clustering results, which is initialized by randomly choosing k samples as centroids.

- *YS algorithm (SpYS)* This refers to the algorithm proposed in Yu and Shi [67]. The clustering results are obtained by finding the optimal orthogonal transformation of $\hat{H} = D^{-1/2}H$ into a partition matrix [67], where columns of H are the k leading eigenvectors of A .

4. *SymNMF* We observed that the Newton-like algorithm for SymNMF gives better clustering quality on image data (more details in Sect. 7.5). On text data, however, the Newton-like algorithm is not efficient enough due to large problem sizes, and only the ANLS algorithm is applicable. When reporting the results, we use the general name “SymNMF” to refer to the algorithm of choice.

For the Newton-like algorithm (Algorithm 1), we use parameters $\beta = 0.1, \sigma = 0.1$. We also empirically observe that choosing ϵ in (12) to be a fixed value 10^{-16} makes the Newton-like algorithm faster while having little influence on the clustering quality. For the ANLS algorithm, we solve the formulation (15), i.e. without sparsity constraints on W, H (Algorithm 2). We empirically observe that it is sufficient to use a fixed parameter $\alpha = 1$ in (15) to obtain a negligible $\|W - H\|_F / \|H\|_F$. Note that the choice of a large enough value of α should be aligned with the scale of the similarity values in A . In our experiments, the matrix A contains normalized similarity values (25), thus the maximum possible value in A is 1, and most of the entries of A are smaller than 1. Finally, in both of our algorithms, the tolerance parameter μ in the stopping criteria is set to 10^{-4} and the maximum iteration count is set to 10,000 so that the outputs are stationary points.

For each data set, we run each algorithm 20 times with different random initializations and the known number of clusters k as input. Algorithms in the same category have the same initializations. In other words, the multistart global optimization method is applied with the described method in each category as for a local search procedure. Although the data sets are labeled, the labels are used only when evaluating the clustering quality, not by the clustering algorithms.

7.4 Clustering quality

We use *clustering accuracy*, the percentage of correctly clustered items given by the maximum bipartite matching, to evaluate the clustering quality (see more details in [64]). The average and maximum clustering accuracy over the 20 runs are shown in Tables 5 and 6, respectively. The maximum clustering accuracy for an algorithm is determined by the solution with the smallest objective function value among the 20 runs. We have the following observations:

1. SpYS and SymNMF achieves the highest clustering quality more frequently than other methods. Note that SpYS was proposed as a more principled way to obtain hard clustering than SpNJW, from the k leading eigenvectors of A [67]. Conceptually, both SpYS and SymNMF facilitate interpretation of the low-rank matrix that is used to approximate the graph similarity matrix, so that we can obtain hard clustering results directly from the low-rank matrix. However, comparing Tables 5 and 6, we observe that by employing the multistart global optimization method and picking the solution with the smallest objective function value, SpYS achieves higher accuracy than the average for four out of eight data sets, while SymNMF and SpNJW achieve higher accuracy than the average for 5 data sets, implying that the objective functions in SymNMF and SpNJW are slightly better proxies for the clustering problem than that in SpYS.

Table 5 Average clustering accuracy for document and image data sets

	KM	SKM	KKM	NMF	GNMF	SpNJW	SpYS	SymNMF
TDT2	0.6711	0.6755	0.6837	0.8505	0.7955	0.7499	0.9050	0.8934
Reuters	0.4111	0.3332	0.3489	0.3731	0.4460	0.3114	0.4986	0.5094
RCV1	0.3111	0.3888	0.3831	0.3797	0.3592	0.2723	0.2743	0.2718
NIPS14-16	0.4602	0.4774	0.4908	0.4918	0.4908	0.4987	0.5026	0.5086
COIL-20	0.6184	0.5611	0.2881	0.6312	0.6304	0.6845	0.7899	0.7258
ORL	0.6499	0.6500	0.6858	0.7020	0.7282	0.7127	0.7752	0.7798
Extended YaleB	0.0944	0.0841	0.1692	0.1926	0.2109	0.1862	0.2254	0.2307
PIE-expression	0.7358	0.7420	0.7575	0.7912	0.8235	0.7966	0.7375	0.7517
<i>ALL</i>	<i>0.4940</i>	<i>0.4890</i>	<i>0.4759</i>	<i>0.5515</i>	<i>0.5606</i>	<i>0.5265</i>	<i>0.5886</i>	<i>0.5839</i>

For each data set, the highest accuracy and any other accuracy within the range of 0.01 from the highest accuracy are marked bold

The average metrics over all the data sets are marked italic

Table 6 Maximum clustering accuracy for document and image data sets

	KM	SKM	KKM	NMF	GNMF	SpNJW	SpYS	SymNMF
TDT2	0.7878	0.7531	0.7502	0.8761	0.8439	0.8046	0.9060	0.9059
Reuters	0.5001	0.3047	0.3828	0.3839	0.4053	0.3096	0.4985	0.4957
RCV1	0.3392	0.3956	0.3844	0.3762	0.3682	0.2695	0.2743	0.2771
NIPS14-16	0.5071	0.4833	0.5048	0.5000	0.4786	0.4952	0.5024	0.5048
COIL-20	0.6917	0.6125	0.3569	0.6653	0.6590	0.7347	0.7986	0.7847
ORL	0.6675	0.6500	0.7125	0.7200	0.7225	0.7700	0.7725	0.7900
Extended YaleB	0.0903	0.0816	0.1785	0.1980	0.2171	0.1864	0.2299	0.2307
PIE-expression	0.7759	0.7586	0.7629	0.7845	0.8190	0.8060	0.7888	0.7543
<i>ALL</i>	<i>0.5450</i>	<i>0.5049</i>	<i>0.5041</i>	<i>0.5630</i>	<i>0.5642</i>	<i>0.5470</i>	<i>0.5964</i>	<i>0.5929</i>

For each data set, the highest accuracy and any other accuracy within the range of 0.01 from the highest accuracy are marked bold

The average metrics over all the data sets are marked italic

- GNMF in our experiments does not show as dramatic improvement over SpNJW as the results reported in [8] where only maximum clustering accuracy was reported. One possible reason is that in [8], full graphs with cosine similarity are used, whereas we use sparse graphs and different similarity measures for better scalability and clustering quality (Sect. 7.1).
- The K-means variants give exceedingly high accuracy on the RCV1 data set. We need more study to have a good explanation of their performances, for example, in what cases cosine dissimilarity is a better choice of distance measure than Euclidean distance. Note that RCV1 is the only data set where spherical K-means has the highest accuracy, and also the only data set where NMF performs better than almost all the other low-rank approximation methods (GNMF, SpNJW, SpYS, SymNMF). This consistency corroborated with our observation that spherical K-means has a closer relationship to NMF than standard K-means, and seems to explain why spherical K-means is often used as an initialization strategy for NMF [63].

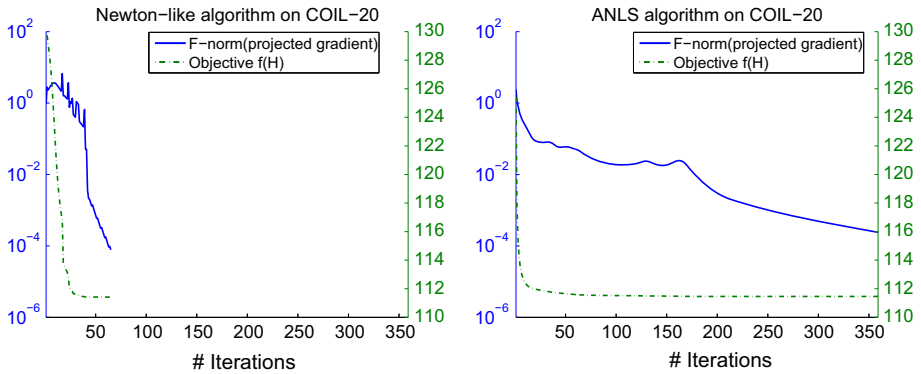


Fig. 7 Convergence behaviors of SymNMF algorithms, generated from a single run on COIL-20 data set with the same initialization

7.5 Convergence and efficiency of SymNMF algorithms

We mentioned in Sect. 7.3 that the ANLS algorithm for SymNMF handles large data sets more efficiently, and the Newton-like algorithm achieves higher clustering accuracy. Here we discuss this tradeoff between efficiency and quality. The different properties exhibited by the two algorithms can be attributed to their different convergence behaviors, though both algorithms converge to stationary point solutions. In Fig. 7, we use COIL-20 data set to study their convergence by plotting the objective function $f(H)$ and the projected gradient $\|\nabla^P f(H)\|_F$ throughout the iterations. As we could expect, $f(H)$ is non-increasing in both algorithms; on the contrary, $\|\nabla^P f(H)\|_F$ is not guaranteed to drop in every iteration but is used to check stationarity.

The Newton-like algorithm shows a divergent behavior in the initial stage of iterations, because the formulation (6) is nonconvex and the search step degrades to a steepest descent direction. However, when the intermediate iterate becomes close to a local minimum, the Hessian matrix becomes positive definite and the second-order information begins to help guide the search. Thus after this point, the algorithm converges very quickly to an accurate stationary point. In contrast, the ANLS algorithm shows a quick drop in both $\|\nabla^P f(H)\|_F$ and $f(H)$ when the algorithm starts. However, near the final stage, it converges slowly to the appointed stationarity level. Overall, the Newton-like algorithm produces more accurate solutions and better clustering quality; however, it is overall less efficient than the ANLS algorithm due to heavier computational cost per iteration. We compare their clustering quality and timing performance in Table 7, with $\mu = 10^{-4}$ in the stopping criterion in both algorithms.

8 Image segmentation experiments

In this section, we explore the application of SymNMF to image segmentation. Image segmentation methods have been heavily relying on spectral clustering [1, 12, 19, 46, 48]. We will demonstrate that SymNMF produces segmentation results that are closer to human-marked boundaries compared to spectral clustering. To the best of our knowledge, this is the first systematic evaluation of SymNMF applied to image segmentation.

Table 7 Clustering accuracy and timing of the Newton-like and ANLS algorithms for SymNMF

	Newton-like algorithm		ANLS algorithm	
	Accuracy	Time (s)	Accuracy	Time (s)
COIL-20	0.7258	53.91	0.7195	8.77
ORL	0.7798	4.30	0.7713	1.97
Extended YaleB	0.2307	163.6	0.2296	23.47
PIE-expression	0.7517	13.48	0.6836	4.43

Experiments are conducted on image data sets with parameter $\mu = 10^{-4}$ and the reported measures are averaged over 20 random runs

8.1 Overview

Image segmentation is an important task in computer vision that organizes an image into a non-overlapping set of closed regions. It can be viewed as a graph clustering problem: The input is a nonnegative and symmetric matrix that contains similarity values between pairs of pixels; the output is a clustering of pixels where each cluster corresponds to a region.

In the graph represented by a pixel similarity matrix A , a pixel is only connected to the pixels within some neighborhood. Thus, the input matrix A is typically a sparse matrix. The similarity value between two neighboring pixels can be computed based on brightness, color, and texture cues [46, 48]. The similarity value characterizes the discontinuity along the line connecting the two pixels and can be trained by a logistic model using human-marked boundaries as ground-truth [19].

Spectral clustering is one of the most common methods that solve the graph clustering problem in image segmentation. As we explained in Sects. 3 and 4, because eigenvectors contain both positive and negative numbers in general, they cannot be used as cluster indicators directly. A variety of methods have been proposed to post-process the graph embedding—the continuous-valued eigenvectors—to obtain closed regions. In contrast, the low-rank matrix H in the solution of SymNMF can not only be used as graph embedding, but also derive graph clustering results directly.

In the current paper, our focus is the gain in segmentation quality by replacing spectral clustering with SymNMF. We follow the steps in an early paper [19] to construct the similarity matrix as well as post-process the graph embedding when the produced low-rank matrix is viewed as graph embedding. The post-processing steps are:

1. Run K-means on the embedded points to generate an oversegmentation of an image. The oversegmentations are called *superpixels* and denoted as o_1, \dots, o_K , where K is an integer larger than the rank k of the low-rank matrix.
2. Build a contracted graph on the superpixels and represent it by a $K \times K$ similarity matrix W . The edge weight between the I -th and J -th superpixels ($1 \leq I, J \leq K$) is defined as:

$$W_{IJ} = \sum_{i \in o_I} \sum_{j \in o_J} A_{ij}. \tag{26}$$

3. Recursively split the contracted graph to produce a hierarchy of regions [46].

We note that the baseline segmentation algorithm [19] used in our comparison between spectral clustering and SymNMF is not the best algorithm to date (for example, see [1]). However, we chose this baseline algorithm in order to simplify the experiment setting and

make the comparison more visible. In our current workflow, both spectral and SymNMF use the same similarity matrix as an input; the resulting low-rank matrices are interpreted as either graph embedding to produce a hierarchy of regions or graph clustering to produce a flat partitioning of an image into regions. With more recent segmentation algorithms such as [1], the low-rank matrices would be interpreted in a more sophisticated way so that we do not know which component of the segmentation algorithm contributes to the gain in segmentation quality. We expect that the comparison result shown in this section will carry on to other segmentation algorithms.

8.2 Data and software

We use the Berkeley Segmentation Data Set 500⁹ (BSDS500) [1] and choose the 200 color images used in [19]. The size of the original images is 481×321 . We resized the images to 240×160 to be consistent with the experiments in [19, 48].

We compute the pixel similarity matrices and post-process the embedded points using the Berkeley Segmentation Engine.¹⁰ We use the default settings: The number of eigenvectors in spectral clustering k (and also the lower rank in SymNMF) is set to 16; the number of oversegmentations K is set to 51. The neighborhood of a pixel is modified from default to a round disk centered at the pixel with radius of 20 pixels. The resulting similarity matrix has size $n \times n$ where $n = 38,400$ and 44 million nonzeros. The same similarity matrix is given as an input to both spectral clustering and SymNMF.

8.3 Evaluation methods

The evaluation of segmentation results is based on the evaluation of boundary detection. In the experiments on document and image clustering, solving SymNMF and interpreting the low-rank result matrix as a cluster indicator yield a hard clustering of items. In order to evaluate SymNMF in the context of image segmentation and compare its performance with that of spectral clustering, we introduce our way to transform the hard clustering results to soft boundaries. First, we generate a probability of boundary (P_b) image from multiple segmentations of an image. Second, we evaluate the P_b image against human-marked boundaries.

– We consider the following four ways to obtain multiple segmentations:

1. **Spectral-Embed** Compute the eigenvectors associated with the 16 largest eigenvalues and treat them as a graph embedding. Generate a hierarchy of regions following the procedures in Sect. 8.1. Each level of the hierarchy determines a segmentation of the image.
2. **SymNMF-Embed** Solve SymNMF with $k = 16$ and treat the rows of H as a graph embedding. Generate a hierarchy of regions following the procedures in Sect. 8.1. Each level of the hierarchy determines a segmentation of the image.
3. **Spectral-NJW** For each $k = 2, 3, \dots, 16$, compute the eigenvectors associated with the k largest eigenvalues, denoted as a matrix $\hat{H} \in \mathbb{R}^{n \times k}$. Apply K-means to the rows of each matrix \hat{H} , and the clustering result corresponds to a segmentation.
4. **SymNMF-Clust** Solve SymNMF with $k = 2, 3, \dots, 16$ and treat each matrix H as a cluster indicator. For each k , the clustering result corresponds to a segmentation. Spectral-Embed and SymNMF-Embed produces 50 segmentations for each image. Spectral-NJW and SymNMF-Clust produces 15 segmentations for each

⁹ <http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html>.

¹⁰ <http://www.cs.berkeley.edu/~fowlkes/BSE/>.

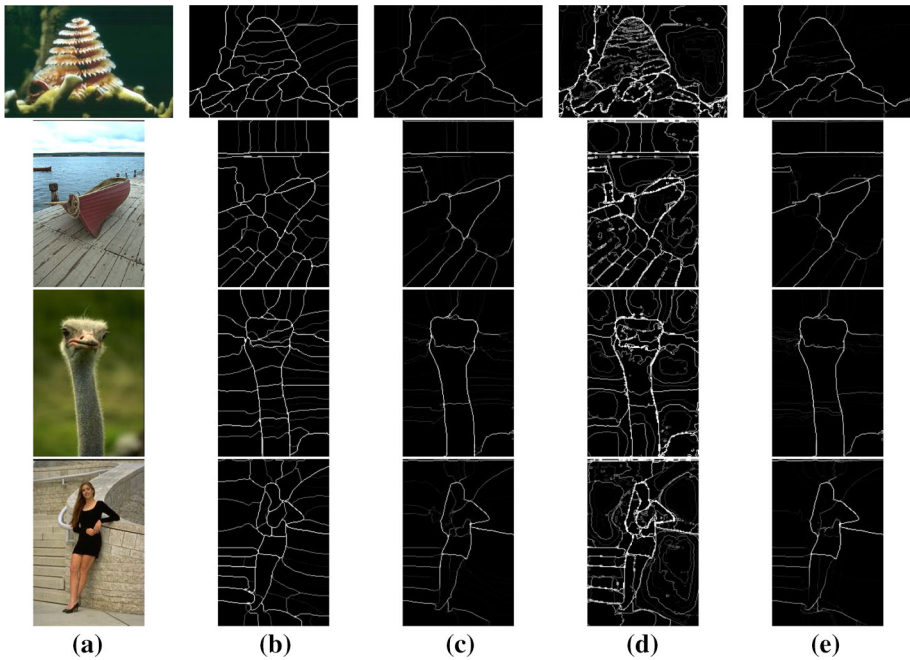


Fig. 8 Examples of the original images and P_b images from BSDS500. Pixels with brighter color in the P_b images have higher probability to be on the boundary. **a** Original, **b** Spectral-Embed, **c** Spectral-NJW, **d** SymNMF-Embed, **e** SymNMF-Clust

image. The P_b value of a pixel is defined as the proportion of times the pixel lies on the boundary determined by the regions in a segmentation. Note that Spectral-NJW and SymNMF-Clust do not enforce hierarchies in their segmentations. Among these four ways of post-processing, only Spectral-Embed was used for evaluation against human-marked boundaries in existing work.

- The data set includes a couple of human-marked boundaries for each image for evaluation. The P_b image has values in the $[0, 1]$ interval. We can produce a binary boundary image using a threshold value t ($0 < t < 1$). Then the *precision* P is calculated as the fraction of true boundary pixels among all the detected boundary pixels; the *recall* R is calculated as the fraction of detected boundary pixels among all the true boundary pixels. The F-measure is defined as $2PR/(P + R)$. We can draw a precision-recall curve using a series of threshold values (see more details in [49]). The best F-measure on this curve is regarded as a summary performance metric.

8.4 Results

We show the precision-recall curves for Spectral-Embed, SymNMF-Embed, Spectral-NJW, and SymNMF-Clust in Fig. 8. Using the best F-measure as the summary metric, both SymNMF versions have better segmentation quality than either of the spectral clustering methods.

SymNMF-Embed is much better than Spectral-Embed in the high-recall low-precision area, with the highest recall approaching 0.8.

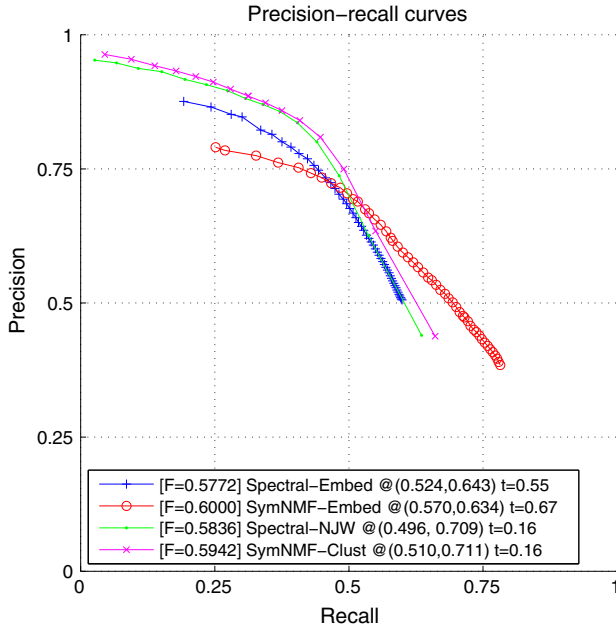


Fig. 9 Precision-recall curves for image segmentation

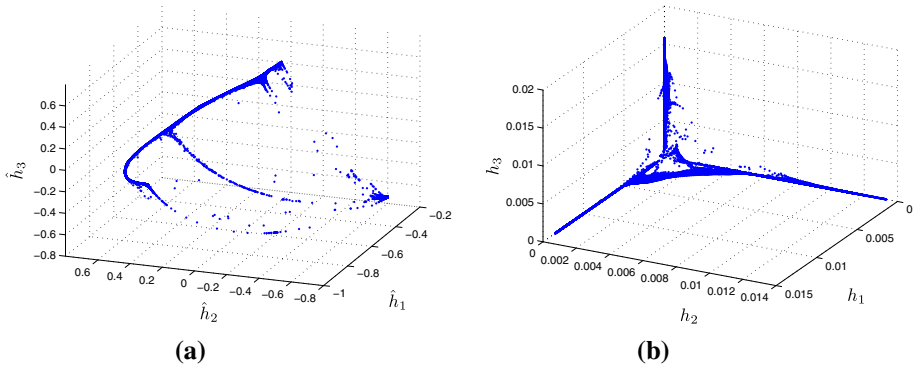


Fig. 10 Illustration of different graph embeddings produced by spectral clustering and SymNMF for the third color image in Fig. 8. **a** The rows of the first three eigenvectors $\hat{H} \in \mathbb{R}^{n \times 3}$ are plotted. **b** The rows of $H \in \mathbb{R}_+^{n \times 3}$ in the result of SymNMF with $k = 3$ are plotted. Each dot corresponds to a pixel. **a** spectral clustering, **b** SymNMF

SymNMF-Clust is much better than Spectral-Embed in the high-precision low-recall area, and consistently better than Spectral-Embed along the curve. When the threshold value t is close to 1, we can be much more confident about the detected regions using SymNMF-Clust than using Spectral-Embed.

SymNMF-Clust is only marginally better than Spectral-NJW, but is consistently better along the precision-recall curve.

Figure 9 shows several exemplar images from the BSDS500 data set. The segmentation results are consistent with our findings in the precision-recall curve. We notice that

Spectral-Embed often subdivides a large flat area with uniform colors into multiple regions (grass, sky, etc.). This is a well-known problem of image segmentation methods that rely on K-means to post-process the eigenvectors, and the reason is that the embedded points for the pixels in those areas vary smoothly [1]. On the contrary, SymNMF-Clust often leaves those areas intact, which implies that the low-rank matrix generated by SymNMF is a better cluster indicator. Figure 10 shows the pixels plotted in the lower dimensional space produced by spectral clustering and SymNMF for a single image, which seems to support our reasoning above. We also notice that SymNMF-Clust tends to identify a few very small regions that correspond to noise in an image. This means that setting k larger than needed will not degrade its segmentation quality. If we remove the regions whose areas are smaller than some threshold, we will see that many of the remaining regions correspond to meaningful objects.

In summary, we can use SymNMF-Clust to detect salient objects and use SymNMF-Embed to discover more detailed segments.

9 Conclusion

In this paper, we studied SymNMF: $\min_{H \geq 0} \|A - HH^T\|_F^2$ as a graph clustering method that is suitable for clustering data points embedded in linear and nonlinear manifolds. Our method extends the applicability of NMF to more general cases, where data relationship is not described by distances in vector space but by similarity values in a latent space. Unlike previous work on SymNMF that imposed various additional constraints on the matrix H , we showed that with nonnegativity constraints only, H can be well interpreted as a cluster indicator matrix. We justified SymNMF to be a valid graph clustering method by showing that it originates from the same formulation as spectral clustering but relaxes the constraint on H differently. While spectral clustering methods require post-processing the eigenvector-based data representation to obtain hard clusters, SymNMF does not depend on the spectrum and finds cluster memberships directly from H . Compared to previous work on the extension of NMF to a positive semi-definite and nonnegative matrix, our approach only assumes that A is symmetric and nonnegative.

We developed two algorithms for SymNMF, a Newton-like algorithm and an ANLS-based algorithm, which should be used in different cases for best practices but both guaranteed to converge to stationary point solutions. We discussed the tradeoff between clustering quality and efficiency when choosing an algorithm for SymNMF. On one hand, the Newton-like algorithm often produces more accurate solutions and higher-quality clustering results, but is more appropriate when the problem size n is small, e.g. $n < 3,000$. On the other hand, the ANLS algorithm is especially efficient for a sparse input matrix A and is scalable to very large data sets, e.g. $n \approx 10^6$. For large-scale clustering, we have to construct a sparse similarity matrix instead of a dense one. For example, with $n = 10^5$ data points, it is difficult to store a dense similarity matrix (~ 75 GB) into the main memory of a contemporary machine.

We have shown the promise of SymNMF in document clustering and image clustering. We also conducted a comprehensive evaluation of SymNMF for image segmentation on 200 natural images. Overall, we developed a *general* framework in this paper, one with minimal constraints and flexible enough for extension. One limitation of our formulation is that an indefinite matrix A could be approximated by a positive semi-definite matrix HH^T . Its effect requires further study; however, we have not seen evidences that the clustering performance degraded due to this limitation. The proposed algorithms can be easily parallelized, for example, in the Newton-like algorithm, the evaluation and Cholesky factorization of different diagonal blocks of the Hessian can run in parallel; and in the ANLS algorithm, the nonnegative least squares problem with different right-hand sides can be made parallel as well.

Acknowledgments The work of the first and third authors was supported in part by the National Science Foundation (NSF) Grants CCF-0808863 and the Defense Advanced Research Projects Agency (DARPA) XDATA program Grant FA8750-12-2-0309. The work of the second author was supported by the TJ Park Science Fellowship of POSCO TJ Park Foundation. 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, the DARPA, or the NRF.

References

1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(5), 898–916 (2011)
2. Arora, R., Gupta, M.R., Kapila, A., Fazel, M.: Clustering by left-stochastic matrix factorization. In: *ICML '11: Proceedings of the 28th International Conference on Machine Learning* (2011)
3. Banerjee, A., Dhillon, I.S., Ghosh, J., Sra, S.: Clustering on the unit hypersphere using von Mises–Fisher distributions. *J. Mach. Learn. Res.* **6**, 1345–1382 (2005)
4. Berman, A., Plemmons, R.J.: *Nonnegative Matrices in the Mathematical Sciences*. SIAM, Philadelphia, PA (1994)
5. Berman, A., Shaked-Monderer, N.: *Completely Positive Matrices*. World Scientific, River Edge, NJ (2003)
6. Bertsekas, D.P.: Projected newton methods for optimization problems with simple constraints. *SIAM J. Control Optim.* **20**(2), 221–246 (1982)
7. Bertsekas, D.P.: *Nonlinear Programming*, 2nd edn. Athena Scientific, Belmont, MA (1999)
8. Cai, D., He, X., Han, J., Huang, T.S.: Graph regularized nonnegative matrix factorization for data representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(8), 1548–1560 (2011)
9. Catral, M., Han, L., Neumann, M., Plemmons, R.J.: On reduced rank nonnegative matrix factorization for symmetric matrices. *Linear Algebra Appl.* **393**, 107–126 (2004)
10. Chan, P., Schlag, M., Zien, J.: Spectral k-way ratio-cut partitioning and clustering. *IEEE Trans. CAD Integr. Circuits Syst.* **13**(9), 1088–1096 (1994)
11. Choo, J., Lee, C., Reddy, C.K., Park, H.: Utopian: User-driven topic modeling based on interactive nonnegative matrix factorization. *IEEE Trans. Vis. Comput. Graph.* **19**(12), 1992–2001 (2013)
12. Cour, T., Benezit, F., Shi, J.: Spectral segmentation with multiscale graph decomposition. In: *CVPR '05: Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1124–1131 (2005)
13. Dhillon, I., Guan, Y., Kulis, B.: A Unified View of Kernel K-Means, Spectral Clustering and Graph Cuts. Technical Report TR-04-25, University of Texas at Austin (2005)
14. Dhillon, I., Modha, D.S.: Concept decompositions for large sparse text data using clustering. *Mach. Learn.* **42**, 143–175 (2001)
15. Ding, C., He, X., Simon, H.D.: On the equivalence of nonnegative matrix factorization and spectral clustering. In: *SDM '05: Proceedings of the SIAM International Conference on Data Mining*, pp. 606–610 (2005)
16. Ding, C., Li, T., Jordan, M.: Nonnegative matrix factorization for combinatorial optimization: spectral clustering, graph matching, and clique finding. In: *ICDM '08: Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 183–192 (2008)
17. Ding, C., Li, T., Jordan, M.I.: Convex and semi-nonnegative matrix factorization. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**(1), 45–55 (2010)
18. Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. Wiley-Interscience, New York (2000)
19. Fowlkes, C., Malik, J.: How Much Does Globalization Help Segmentation? Technical Report UCB/CSD-4-1340. University of California, Berkeley (2004)
20. Gillis, N., Kuang, D., Park, H.: Hierarchical clustering of hyperspectral images using rank-two nonnegative matrix factorization. *IEEE Trans. Geosci. Remote Sens.* **53**(4):2066–2078, (2015)
21. Globerson, A., Chechik, G., Pereira, F., Tishby, N.: Euclidean embedding of co-occurrence data. *J. Mach. Learn. Res.* **8**, 2265–2295 (2007)
22. Gonzales, E.F., Zhang, Y.: Accelerating the Lee–Seung Algorithm for Non-Negative Matrix Factorization. Technical Report TR05-02, Rice University (2005)
23. He, Z., Xie, S., Zdunek, R., Zhou, G., Cichocki, A.: Symmetric nonnegative matrix factorization: algorithms and applications to probabilistic clustering. *IEEE Trans. Neural Netw.* **22**(12), 2117–2131 (2011)
24. Ho, N.D.: *Nonnegative Matrix Factorization Algorithms and Applications*. Ph.D. thesis, Université catholique de Louvain (2008)

25. Kannan, R., Ishteva, M., Park, H.: Bounded matrix factorization for recommender system. *Knowl. Inf. Syst.* **39**(3), 491–511 (2014)
26. Kelley, C.T.: *Iterative Methods for Linear and Nonlinear Equations*. SIAM, Philadelphia, PA (1995)
27. Kim, H., Park, H.: Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics* **23**(12), 1495–1502 (2007)
28. Kim, H., Park, H.: Nonnegative matrix factorization based on alternating non-negativity-constrained least squares and the active set method. *SIAM J. Matrix. Anal. Appl.* **30**(2), 713–730 (2008)
29. Kim, J., He, Y., Park, H.: Algorithms for nonnegative matrix and tensor factorizations: a unified view based on block coordinate descent framework. *J. Glob. Optim.* **58**(2), 285–319 (2014)
30. Kim, J., Park, H.: Sparse Nonnegative Matrix Factorization for Clustering. Technical Report GT-CSE-08-01, Georgia Institute of Technology (2008)
31. Kim, J., Park, H.: Toward faster nonnegative matrix factorization: a new algorithm and comparisons. In: *ICDM '08: Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 353–362 (2008)
32. Kim, J., Park, H.: Fast nonnegative matrix factorization: An active-set-like method and comparisons. *SIAM J. Sci. Comput.* **33**(6), 3261–3281 (2011)
33. Kleinberg, J.: An impossibility theorem for clustering. *Adv. Neural Inf. Process. Syst.* **15**, 446–453 (2002)
34. Kuang, D., Ding, C., Park, H.: Symmetric nonnegative matrix factorization for graph clustering. In: *SDM '12: Proceedings of the SIAM International Conference on Data Mining*, pp. 106–117 (2012)
35. Kuang, D., Park, H.: Fast rank-2 nonnegative matrix factorization for hierarchical document clustering. In: *KDD '13: Proceedings of the 19th ACM International Conference on Knowledge Discovery and Data Mining*, pp. 739–747 (2013)
36. Kulis, B., Basu, S., Dhillon, I., Mooney, R.: Semi-supervised graph clustering: a kernel approach. In: *ICML '05: Proceedings of the 22nd International Conference on Machine Learning*, pp. 457–464 (2005)
37. Lawson, C.L., Hanson, R.J.: *Solving Least Squares Problems*. Prentice-Hall, Englewood Cliffs, NJ (1974)
38. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. *Nature* **401**, 788–791 (1999)
39. Lee, K.C., Ho, J., Kriegman, D.: Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(5), 684–698 (2005)
40. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.* **5**, 361–397 (2004)
41. Li, T., Ding, C., Jordan, M.I.: Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In: *ICDM '07: Proceedings of the 7th IEEE International Conference on Data Mining*, pp. 577–582 (2007)
42. Lin, C.J.: On the convergence of multiplicative update algorithms for nonnegative matrix factorization. *Trans. Neural Netw.* **18**(6), 1589–1596 (2007)
43. Lin, C.J.: Projected gradient methods for nonnegative matrix factorization. *Neural Comput.* **19**(10), 2756–2779 (2007)
44. Lucena, A., Ribeiro, C., Santos, A.C.: A hybrid heuristic for the diameter constrained minimum spanning tree problem. *J. Glob. Optim.* **46**(3), 363–381 (2010)
45. Ma, X., Gao, L., Yong, X., Fu, L.: Semi-supervised clustering algorithm for community structure detection in complex networks. *Phys. A Stat. Mech. Appl.* **389**(1), 187–197 (2010)
46. Malik, J., Belongie, S., Leung, T., Shi, J.: Contour and texture analysis for image segmentation. *Int. J. Comput. Vis.* **43**(1), 7–27 (2001)
47. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press, New York, NY (2008)
48. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(5), 530–549 (2004)
49. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: *ICCV '01: Proceedings of the 8th IEEE International Conference on Computer Vision*, vol. 2, pp. 416–423 (2001)
50. Nepusz, T., Petróczy, A., Négyessy, L., Bazsó, F.: Fuzzy communities and the concept of bridgeness in complex networks. *Phys. Rev. E* **77**(1), 016,107 (2008)
51. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. *Adv. Neural Inf. Process. Syst.* **14**, 849–856 (2001)
52. Pauca, V.P., Shahnaz, F., Berry, M.W., Plemmons, R.J.: Text mining using non-negative matrix factorizations. In: *SDM '04: Proceedings of the SIAM International Conference on Data Mining*, pp. 452–456 (2004)
53. Rinnooy Kan, A., Timmer, G.: Stochastic global optimization methods, part II: multi level methods. *Math. Program.* **39**, 57–78 (1987)

54. Rinnooy Kan, A., Timmer, G.: Global optimization. In: Kan, R., Todds, M.J. (eds.) *Handbooks in Operations Research and Management Science*, vol. 1, pp. 631–662. North Holland, Amsterdam (1989)
55. Shahnaz, F., Berry, M.W., Pauca, V.P., Plemmons, R.J.: Document clustering using nonnegative matrix factorization. *Inf. Process. Manag.* **42**, 373–386 (2006)
56. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8), 888–905 (2000)
57. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression database. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(12), 1615–1618 (2003)
58. Stewart, G.W., Sun, J.G.: *Matrix Perturbation Theory*. Academic Press, New York (1990)
59. Strogatz, S.H.: Exploring complex networks. *Nature* **410**, 268–276 (2001)
60. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 2579–2605 (2008)
61. von Luxburg, U.: A tutorial on spectral clustering. *Stat. Comput.* **17**(4), 395–416 (2007)
62. Wang, F., Li, T., Wang, X., Zhu, S., Ding, C.: Community discovery using nonnegative matrix factorization. *Data Min. Knowl. Discov.* **22**(3), 493–521 (2011)
63. Wild, S., Curry, J., Dougherty, A.: Improving non-negative matrix factorizations through structured initialization. *Pattern Recognit.* **37**, 2217–2232 (2004)
64. Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: *SIGIR '03: Proceedings of the 26th International ACM Conference on Research and Development in Information Retrieval*, pp. 267–273 (2003)
65. Yang, Z., Hao, T., Dikmen, O., Chen, X., Oja, E.: Clustering by nonnegative matrix factorization using graph random walk. *Adv. Neural Inf. Process. Syst.* **25**, 1088–1096 (2012)
66. Yang, Z., Oja, E.: Clustering by low-rank doubly stochastic matrix decomposition. In: *ICML '12: Proceedings of the 29th International Conference on Machine Learning* (2012)
67. Yu, S.X., Shi, J.: Multiclass spectral clustering. In: *ICCV '03: Proceedings of the 9th IEEE International Conference on Computer Vision*, pp. 313–319 (2003)
68. Yun, S., Tseng, P., Toh, K.C.: A block coordinate gradient descent method for regularized convex separable optimization and covariance selection. *Math. Program.* **129**, 331–355 (2011)
69. Zass, R., Shashua, A.: A unifying approach to hard and probabilistic clustering. In: *ICCV '05: Proceedings of the 10th IEEE International Conference on Computer Vision*, pp. 294–301 (2005)
70. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. *Adv. Neural Inf. Process. Syst.* **17**, 1601–1608 (2004)
71. Zhang, Y., Yeung, D.Y.: Overlapping community detection via bounded nonnegative matrix tri-factorization. In: *KDD '12: Proceedings of the 18th ACM International Conference on Knowledge Discovery and Data Mining*, pp. 606–614 (2012)
72. Zhang, Z.Y., Wang, Y., Ahn, Y.Y.: Overlapping community detection in complex networks using symmetric binary matrix factorization. *Phys. Rev. E* **87**(6), 062,803 (2013)