Symmetric Nonnegative Matrix Factorization for Graph Clustering

Da Kuang* Chris Ding† Haesun Park*

Abstract

Nonnegative matrix factorization (NMF) provides a lower rank approximation of a nonnegative matrix, and has been successfully used as a clustering method. In this paper, we offer some conceptual understanding for the capabilities and shortcomings of NMF as a clustering method. Then, we propose Symmetric NMF (SymNMF) as a general framework for graph clustering, which inherits the advantages of NMF by enforcing nonnegativity on the clustering assignment matrix. Unlike NMF, however, SymNMF is based on a similarity measure between data points, and factorizes a symmetric matrix containing pairwise similarity values (not necessarily nonnegative). We compare SymNMF with the widely-used spectral clustering methods, and give an intuitive explanation of why SymNMF captures the cluster structure embedded in the graph representation more naturally. In addition, we develop a Newton-like algorithm that exploits second-order information efficiently, so as to show the feasibility of SymNMF as a practical framework for graph clustering. Our experiments on artificial graph data, text data, and image data demonstrate the substantially enhanced clustering quality of SymNMF over spectral clustering and NMF. Therefore, SymNMF is able to achieve better clustering results on both linear and nonlinear manifolds, and serves as a potential basis for many extensions and applications.

1 Introduction

In nonnegative matrix factorization (NMF), given a nonnegative matrix X, and a reduced rank k, we seek a lower-rank matrix approximation given by

$$X \approx CG^T \tag{1.1}$$

Using Forbenius norm to measure the distance between X and CG^T , the problem of computing NMF is formulated as [11]:

$$\min_{C,G\geq 0} \|X - CG^T\|_F^2 \tag{1.2}$$

where $X \in \mathbb{R}_+^{m\times n}, C \in \mathbb{R}_+^{m\times k}, G \in \mathbb{R}_+^{n\times k}$, $\mathbb{R}_+$ denotes the set of nonnegative real numbers, and $\|\cdot\|_F$ denotes Frobenius norm. In general $k < \min\{m, n\}$ and typically k is assumed to be much smaller than m or n, thus NMF represents a lower rank approximation.

The formulation of NMF in (1.2) is easily related to the clustering of nonnegative data [12]. Suppose we have n data points represented as the columns in $X = [x_1, \cdots, x_n]$, and try to group them into k clusters. The result of the well-known K-means clustering can be represented as in (1.2) where the columns of C are the cluster centroids and the i-th column of G^T is e_j if x_i belongs to cluster j (e_j denotes the j-th column of $I_{k\times k}$). Likewise, when C, G are only subject to nonnegativity, we can still interpret the dimension reduction in (1.2) as clustering results: The columns of the first factor C provide the basis of a latent k-dimensional space, and the columns of the second factor G^T provide the representation of $x_1, \cdots, x_n$ in the latent space. Due to nonnegativity, NMF offers the *interpretability* that the clustering assignment of each data point can be easily obtained by choosing the largest entry in the corresponding column of G^T. Note that this is not possible in lower-rank approximation methods with no nonnegativity constraints, such as singular value decomposition (SVD). NMF has received wide attention in clustering with many types of data, including documents [22], images [3], and microarray data [10].

Although NMF has been shown to be effective to perform clustering, the goals of clustering and dimension reduction are different. While a dimension reduction method uses a few basis vectors that well approximate the data matrix, the goal of clustering is to find a partitioning of the data points where *similarity* is high within each cluster and low across clusters. The similarity measure should be defined based on the inherent cluster structure. When these two goals coincide, i.e. a basis vector is a suitable representation of one cluster, NMF is able to achieve good clustering results. However, this assumption is violated when the data have

*School of Computational Science and Engineering, Georgia Institute of Technology ({da.kuang,hpark}@cc.gatech.edu). This work was supported in part by the National Science Foundation grants CCF-0732318 and CCF-0808863. 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 National Science Foundation.

†Department of Computer Science and Engineering, University of Texas, Arlington (chqding@uta.edu).

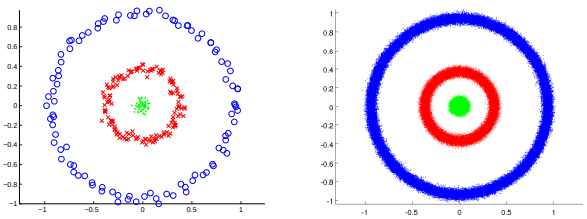
nonlinear cluster structures. It is interesting to note that most success of NMF in clustering has been around document clustering [22, 14, 12]. One reason is that each basis vector represents the word distribution of a topic, and the documents with similar word distributions should be classified into the same group. This property does not hold in every type of data. For example, it was shown that a collection of images tends to form multiple 1-dimensional nonlinear manifolds [19].

The standard formulation of NMF in (1.2) has been applied to many clustering tasks where the n data points are explicitly available in X and are directly used as input. However, in many cases, such as when data points are embedded in a nonlinear manifold, it is better to describe the relationship between data points in the form of a graph. In the graph model, each node corresponds to a data point, and a *similarity matrix* $A_{n\times n}$ contains similarity values between each pair of nodes, i.e. the (i,j)-th entry of A represents the similarity between x_i and x_j. In this paper, we explore a symmetric variation of NMF that uses A directly as input. When A is properly constructed, the factorization of A will generate a clustering assignment matrix that is nonnegative and well captures the cluster structure inherent in the graph representation. This way, we propose Symmetric NMF (SymNMF) as a new method for graph clustering. We expect that SymNMF can inherit the good interpretability of NMF. Note that a similarity matrix A is not required to be nonnegative; however, we do emphasize the nonnegativity of the clustering assignment matrix SymNMF produces.

We can show that our formulation of SymNMF is related to a group of widely-used graph clustering methods, namely *spectral clustering*. In fact, Ding et al. [7] made an important observation that relates NMF and spectral clustering via the objective function of kernel K-means:

$$\min_{H^TH=I, H\geq 0} \|X^TX - HH^T\|_F^2 \tag{1.3}$$

where X is a data matrix defined in (1.2). X^TX can also be extended to a positive semi-definite kernel matrix $K_{n\times n}$. They also proposed an algorithm for the factorization of a nonnegative kernel matrix. However, in general, a similarity matrix A in graph clustering is neither nonnegative nor positive semi-definite. In this respect, SymNMF can be used to factorize a larger class of symmetric matrices that contain pairwise similarity values, thus has a closer relationship to spectral clustering and is well justified to be a graph clustering method. We will show that SymNMF and spectral clustering try to solve the same optimization problem by relaxing the constraints in different ways. However, the nonnegativity property ($H \geq 0$) retained in SymNMF is crucial for its success since the interpretation of H into clustering results is straightforward, i.e. by picking the largest entry in each row of H, as in the case of NMF. We will show the uniqueness of SymNMF in terms of clustering quality, the independence on the eigenspace of A, and the sensitiveness to the fluctuation in the similarity matrix A. We will also develop an algorithm for SymNMF that is guaranteed to produce stationary point solutions.



(a) 300 data points. (b) 300,000 data points.

Figure 1: (a) A graph clustering example with 3 clusters (reproduced from [24]); (b) Another example with similar cluster structure and a much larger number of data points.

The rest of the paper is organized as follows. In Section 2, we present the formulation of SymNMF and argue that it has additional clustering capability compared to NMF and meanwhile has good interpretability offered by nonnegativity. We also explain the difference between SymNMF and spectral clustering in terms of the inherent cluster structure they may capture. In Section 3, we develop a Newton-like algorithm for SymNMF that exploits the second-order information efficiently. It has better convergence rate than simple gradient descent methods that relies on the first-order information only. In Section 4, we demonstrate the superiority of SymNMF and the Newton-like algorithm with experiments on artificial graph data, text data, and image data. SymNMF has better clustering quality than NMF because it offers a wider range of similarity measures to choose from. Experiments also show that SymNMF is better than spectral clustering because it naturally captures the cluster structure and does not require additional steps to obtain clustering results. In Section 5, we discuss related work on nonnegative factorization of a symmetric matrix. In Section 6, we summarize the benefits of this new clustering framework and give comments on future research directions.

2 Symmetric NMF (SymNMF)

2.1 SymNMF and NMF Although NMF has been widely used in clustering and often reported to have better clustering quality than classical methods such as

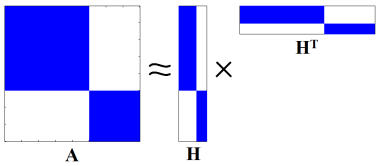


Figure 2: An illustration of the symmetric factorization of a similarity matrix: $A \approx HH^T$. Regions with dark colors indicate matrix entries of large value, and white regions indicate entries of small value. The rows and columns of A have been reorganized to place data points in the same group continuously.

K-means, it is not a general clustering method that performs well in every circumstance. The reason is that the clustering capability of an algorithm and its limitation can be attributed to its assumption on the cluster structure. NMF assumes that *each cluster can be represented by a single basis vector, and different clusters must correspond to different basis vectors.* This can be seen from the way to use G^T (the representation of X in a lower-dimensional space) to infer clustering assignments.

The limitations of NMF comes from the fact that its goal is to approximate the original data matrix (using a linear combination of basis vectors). When the underlying k clusters have nonlinear structure and lie on a nonlinear manifold (for example, see Fig. 1a), NMF cannot find any k basis vectors that represent the k clusters respectively. On the other hand, there is another important class of clustering methods, which views the data points in a graph model and minimizes some form of graph cuts. Although a graph model is commonly built upon coordinates of data points in their original Euclidean space, the coordinate information no longer plays a role after the graph is constructed based on the similarity between each pair of data points. Thus, clustering methods belonging to this class are only concerned with an $n \times n$ similarity matrix A, where n is the number of data points or nodes. Extending the interpretability of NMF to this symmetric case is the focus of this section.

We formulate the nonnegative symmetric factorization (SymNMF) of the similarity matrix A as:

$$\min_{H \geq 0} \|A - HH^T\|_F^2 \tag{2.4}$$

where H is a nonnegative matrix of size $n \times k$, and k is the number of clusters requested. The effectiveness of this formulation is conceptually depicted in Fig. 2. Assume data points in the same group have large similarity values, and data points in different groups have small similarity values. Then a good approximation to A naturally captures the cluster structure, and the largest entry in the i-th row of H indicates the clustering assignment of the i-th data point due to the nonnegativity of H.

Compared to NMF, SymNMF is more flexible in terms of choosing similarities for the data points. We can choose whatever similarity measure that well describes the inherent cluster structure. In fact, the formulation of NMF (1.2) can be related to SymNMF when $A = X^T X$ in the formulation (2.4) [7]. This means that NMF implicitly chooses inner products as the similarity measure, which might not be suitable to distinguish different clusters.

In previous work on symmetric NMF [7, 4], A is required to be nonnegative *and* symmetric positive semidefinite. Our formulation differs in that the matrix A in (2.4) can be any symmetric matrix representing similarity values. A similarity value may be a negative number (for example, in some definition of kernel matrices used in semi-supervised clustering [13]). However, in the SymNMF formulation, if A has entries of mixed signs, the magnitudes of negative entries of A do not affect the optimal solution of (2.4). For example, suppose $A_{ij} = p < 0$ and a global optimal solution of (2.4) is H_{opt}. It can be shown that if A_{ij} is changed to any real number $\tilde{p} \leq 0$ while other entries of A are unchanged, then H_{opt} is still a global optimal solution. Therefore, we suggest transforming a similarity matrix with mixed signs into a nonnegative matrix before applying SymNMF.

2.2 SymNMF and Graph Clustering Objectives We can show that the formulation (2.4) is connected to an optimization problem generalized from many graph clustering objective functions; and by doing so, we can solve the remaining question: how to have a proper and theoretically sound construction of the similarity matrix A. This way, SymNMF can be seen as a new algorithm to solve many graph clustering problems.

First, we introduce the notations for a rigorous description of graphs. A graph $G = (V, E)$ is formed by modeling each data point as a node in V, and two nodes can be connected by an edge in E. The goal of graph clustering is to partition the node set V into k nonoverlapping subsets $V_1, \cdots, V_k$. Suppose we have an affinity matrix A^G, where A^G_{ij} is the edge weight between nodes i and j ($A^G_{ij} = 0$ if they are not connected). Note that we call A^G as an *affinity matrix* of a graph, different from the *similarity matrix* A that is used as the matrix to be factorized in the SymNMF formulation (2.4). Define the degree of node i: $d_i = \sum_{j=1}^n A^G_{ij}$, and the diagonal matrix $D = \text{diag}(d_1, \cdots, d_n)$.

Table 1 lists three examples of different graph cut

Table 2: Algorithmic steps of spectral clustering and SymNMF clustering

	Spectral clustering	SymNMF
Objective	$\min_{H^TH=I} \|A - HH^T\|_F^2$	$\min_{H\geq 0} \|A - HH^T\|_F^2$
Step 1	Obtain the global optimal $H_{n\times k}$ by computing k leading eigenvectors of A	Obtain a stationary point solution using some minimization algorithm
Step 2	Normalize each row of H	(no need to normalize H)
Step 3	Infer clustering assignments from the rows of H (e.g. by K-means)	The largest entry in each row of H indicates the clustering assignments

Table 1: Graph clustering objectives and similarity matrices.

Ratio association	Objective: $\max \sum_{p=1}^{k} \frac{\sum_{i\in V_p, j\in V_p} A_{ij}^G}{\|V\|}$ Similarity matrix: $A = A^G$
Normalized cut	Objective: $\max \sum_{p=1}^{k} \frac{\sum_{i\in V_p, j\in V_p} A_{ij}^G}{\sum_{i\in V_p, j\in V} A_{ij}^G}$ Similarity matrix: $A = D^{-1/2}A^GD^{-1/2}$
Kernel clustering	Similarity matrix: $A = K$ $\left[K_{ij} = \phi(x_i)^T\phi(x_j)\right]$

objective functions that have been shown effective in graph clustering, as well as the similarity matrices A. All the cases in Table 1 can be reduced to the following trace maximization form, assuming the similarity matrix A is constructed correspondingly [5, 13]:

$$\max \text{trace}(H^TAH) \tag{2.5}$$

where $H \in \mathbb{R}^{n\times k}$ satisfies some constraints and indicates the clustering assignment. A group of successful graph clustering methods – spectral clustering, relax the constraints on H to $H^TH = I$. Under such orthogonality constraint on H, we have [7]:

$$\begin{aligned} & \max \text{trace}(H^TAH) \\ \Leftrightarrow\ & \min \text{trace}(A^TA) - 2\text{trace}(H^TAH) + \text{trace}(I) \\ \Leftrightarrow\ & \min \text{trace}[(A - HH^T)^T(A - HH^T)] \\ \Leftrightarrow\ & \min \|A - HH^T\|_F^2 \end{aligned}$$

Therefore, compared to spectral clustering, SymNMF can be seen as a different relaxation to $\min \|A - HH^T\|_F^2$, i.e. relaxing the constraints on H to be $H \geq 0$. In addition, we can use the same way of constructing the similarity matrix A as in Table 1. The choice of the similarity matrix A depends on the data set and also will influence the clustering results. The 1st and 2nd cases in Table 1 are derived from two different graph cut definitions, where normalized cut often performs better than ratio association [17]. In the 3rd case, K is a kernel matrix where ϕ is a nonlinear mapping of the original data points. Although it was shown that kernel clustering has equivalent objective functions with weighted graph clustering [5], similarity values are defined by inner products and no graph is explicitly constructed. In all of our experiments in this paper, we explicitly construct graphs from the data points; therefore, we choose normalized cut objective and the corresponding similarity matrix $A = D^{-1/2}A^GD^{-1/2}$. The most suitable way to form the similarity matrix depends on the underlying data set, and that is not the focus of this paper. However, we should keep in mind that we are developing a framework for graph clustering that can be applied to a similarity matrix A derived from any graph clustering objective.

2.3 SymNMF and Spectral Clustering Methods Due to different constraints on H, spectral clustering and SymNMF have different properties in the clustering results they generate. Before analyzing their properties, we first compare their algorithmic steps in Table 2. Spectral clustering leads to eigenvector-based solutions of H, which are not necessarily nonnegative; and K-means or more advanced procedures have to be adopted in order to obtain the final clustering assignments. In contrast, the solution found by SymNMF naturally captures the cluster structure. It also indicates the clustering assignments without additional clustering procedures, which heavily depends on initialization, such as K-means.

Spectral clustering is a well-established framework for graph clustering. However, its success relies on the properties of the leading eigenvalues and eigenvectors of the similarity matrix A. It was pointed out by Ng et al. [16], 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 intuitively that the absence of this property will cause

Table 3: Leading eigenvalues of similarity matrices based on Fig. 1.

	$n = 300$	$n = 300,000$
1st	1.000	1.00000000
2nd	1.000	1.00000000
3rd	1.000	1.00000000
4th	0.994	0.99999999
5th	0.993	0.99999997
6th	0.988	0.99999994

spectral clustering to fail due to noise or numerical artifacts. Suppose A contains $k = 3$ diagonal blocks [16], corresponding to 3 clusters:

$$A = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{bmatrix} \tag{2.6}$$

It follows that a set of linearly independent eigenvectors of A can be formed by padding zeros to the eigenvectors of A_1, A_2, A_3. If we construct the similarity matrix A as in normalized cut (Table 1), 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 leading 3 eigenvectors of A can be found by padding zeros to the leading eigenvector of each A_i, and correspond to the 3 ground-truth clusters respectively. However, when $\lambda_2(A_1)$ is so close to 1 that it cannot be distinguished from $\lambda_1(A_1)$ by an eigensolver in finite precision arithmetic, it is possible that the computed eigenvalues satisfy (in exact arithmetic) $\lambda_1(A_1) > \lambda_2(A_1) > \lambda_1(A_2) > \lambda_1(A_3)$. In this case, two clusters are identified within the 1st ground-truth cluster, and the 3rd ground-truth cluster is ignored. For example, we construct similarity matrices using two graph clustering examples shown in Fig. 1 and using normalized cut as the objective function (the data points in each example are transformed into a sparse graph in a principled way as described in Section 4.1). The leading eigenvalues of A in each example are listed in Table 3. We can see that the 3rd and 4th largest eigenvalues are very close when the number of data points $n = 300,000$, and will become indistinguishable as n continues to increase or if we use single precision instead.

On the other hand, the solution of SymNMF is not based on eigenvectors, thus its algorithm does not suffer from potential numerical artifacts in the leading eigenvalues.

3 A Newton-like Algorithm for SymNMF

In this section, we focus on solving SymNMF formulated as a minimization problem (2.4). The objective function in (2.4) 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 [2]. We can directly apply standard gradient search algorithms, which lead to stationary point solutions; however, they suffer from either slow convergence rate or expensive computation cost. Thus, it is not a trivial problem to minimize (2.4) efficiently.

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

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 $0 < \mu << 1$
Initialize x, $x^{(0)} \leftarrow x$
repeat
  Compute scaling matrix S
  Step size $\alpha = 1$
  while true **do**
    $x_{\text{new}} = [x - \alpha S \nabla f(x)]^+$
    if $f(x_{\text{new}}) - f(x) \leq \sigma \nabla f(x)^T (x_{\text{new}} - x)$ **then**
      break
    end if
    $\alpha \leftarrow \beta \alpha$
  end while
  $x \leftarrow x_{\text{new}}$
until $\|\nabla^P f(x)\| \leq \mu \|\nabla^P f(x^{(0)})\|$ [15]
Output: x

3.1 Algorithm Framework First, we introduce several notations for clarity. Let $H = [h_1, \cdots, h_k]$. A vector x of length nk is used to represent the vectorization of H by column, i.e. $x = \text{vec}(H) = [h_1^T, \cdots, h_k^T]^T$. For simplicity, functions applied on x have the same notation with functions applied on H, i.e. $f(x) \equiv f(H)$. $[\cdot]^+$ denotes the projection to the nonnegative orthant, i.e. changing 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 both the numbers of rows and columns of M are multiples of n. $M \succ 0$ refers to positive definiteness of M.

Algorithm 1 describes a framework of gradient search algorithms applied to SymNMF, upon which we will develop our Newton-like algorithm. This description does not specify iteration indices, but updates x

Table 4: Comparison of PGD and PNewton for solving $\min_{H\geq 0} \|A - HH^T\|_F^2$, where H is an $n \times k$ matrix.

Projected gradient descent (PGD)	Projected Newton (PNewton)
Scaling matrix: $S^{(t)} = I_{nk\times nk}$	Scaling matrix: $S^{(t)} = \left(\nabla^2_{\mathcal{E}} f(x^{(t)})\right)^{-1}$
Linear convergence; Zigzagging	Quadratic convergence
$O(n^2k)$ / iteration	$O(n^3k^3)$ / iteration

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 4 lists two choices of $S^{(t)}$ that lead to different gradient search algorithms: projected gradient descent (PGD) [15] and projected Newton (PNewton) [2].

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 [2]. 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 [1] [2]

$$(\nabla^2_R f(x))_{ij} = \begin{cases} \delta_{ij}, & \text{if } i \in R \text{ or } j \in R \\ \left(\nabla^2 f(x)\right)_{ij}, & \text{otherwise} \end{cases} \tag{3.7}$$

We introduce the definition of an index set $\mathcal{E}$ that helps to prove the convergence of Algorithm 1 [2]:

$$\mathcal{E} = \{i | 0 \leq x_i \leq \epsilon, (\nabla f(x))_i > 0\} \tag{3.8}$$

where ϵ depends on x and is usually small ($0 < \epsilon < 0.01$) [9]. In PNewton, $S^{(t)}$ is formed based on the reduced Hessian $\nabla^2_{\mathcal{E}} f(x^{(t)})$ 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 very large computational complexity – $O(n^3k^3)$, which is prohibitive. Therefore, we need to develop a Newton-like algorithm that exploits second-order information in an inexpensive way.

[1] δ_{ij} is the Kronecker delta.

[2] Both the gradient and the Hessian of $f(x)$ can be computed analytically:
$\nabla f(x) = \text{vec}(4(HH^T - A)H)$
$(\nabla^2 f(x))_{[i][j]} = 4\left(\delta_{ij}(HH^T - A) + h_j h_i^T + (h_i^T h_j) I_{n\times n}\right)$

3.2 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 3.1. *A scaling matrix S is diagonal with respect to an index set R, if $S_{ij} = 0, \forall i \in R$ and $j \neq i$. [1]*

LEMMA 3.1. *Let S be a positive definite matrix which is diagonal with respect to $\mathcal{E}$. Then if $x \geq 0$ is not a stationary point, there exists $\bar{\alpha} > 0$ such that $f\left([x - \alpha S \nabla f(x)]^+\right) < f(x), \forall 0 < \alpha < \bar{\alpha}$. (modified from [1])*

Lemma 3.1 states the requirement on $S^{(t)}$, which is satisfied by the choices of $S^{(t)}$ in both PGD and PNewton. It will guide our development of new ways to choose $S^{(t)}$.

3.2.1 Improvement 1 (I1): 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) [9]. 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 $\mathcal{E}^{(t)} \neq \mathcal{E}^{(1)}$, 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,

$$S^{(t)} = \begin{cases} S^{(t-1)}, & \text{if } \mathcal{E}^{(t)} = \mathcal{E}^{(t-1)} \\ \left(\nabla^2_{\mathcal{E}} f(x^{(t)})\right)^{-1}, & \text{if } \mathcal{E}^{(t)} \neq \mathcal{E}^{(t-1)} \\ & \quad \text{and } \nabla^2_{\mathcal{E}} f(x^{(t)}) \succ 0 \\ I_{nk\times nk}, & \text{otherwise} \end{cases} \tag{3.9}$$

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

3.2.2 Improvement 2 (I2): Cheaper Hessian Evaluations The second improvement on choosing $S^{(t)}$ is inspired by the recently proposed *coordinate gradient descent* (CGD) method for solving covariance selection [23]. 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. More precisely,

(3.10)
$$S^{(t)}_{[i][j]} = \begin{cases} 0, \text{ if } i \neq j \\ \left(\nabla^2_{\mathcal{E}} f(x^{(t)})_{[i][j]}\right)^{-1}, \text{if } i = j \\ \qquad\qquad \text{and } \nabla^2_{\mathcal{E}} f(x^{(t)})_{[i][j]} \succ 0 \\ I_{n\times n}, \text{ otherwise} \end{cases}$$

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: 1. The computational complexity in each iteration is $O(n^3k)$, much lower than the complexity of PNewton; 2. We can exploit partial second-order information even though the n diagonal blocks of $\nabla^2_{\mathcal{E}} f(x^{(t)})$ 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 (2.4) is to combine Improvement 1 ($I1$) and Improvement 2 ($I2$). Note that the requirement on $S^{(t)}$ described in Lemma 1 is satisfied in both $I1$ and $I2$, and also in their combination $I1 + I2$. Thus, convergence is guaranteed in all of these variations.

4 Experiments

In this section, we first review how to transform a group of data points into graphs for clustering. With a graph constructed and its affinity matrix A^G available, the similarity matrix A used in the SymNMF formulation (2.4) can be formed using normalized cut as the graph clustering objective function (see Table 1, and details on the choice of similarity matrix in Section 2.2). We then evaluate the four variations of Newton-like algorithms for SymNMF on artificial data, and extend the experiments of the best-performing SymNMF algorithm to real-world data sets.

Throughout the experiments, we use Matlab 7.9 (R2009b) with an Intel Xeon X5550 quad-core processor and 24GB memory. All the Newton-like algorithms following the framework in Algorithm 1 have parameters $\beta = 0.1, \sigma = 0.1, \mu = 10^{-4}$, and the maximum iteration count is set to 10000. In addition, we empirically observe that choosing ϵ in (3.8) to be a fixed value 10^{-16} makes the Newton-like algorithms faster while having little influence on the clustering quality.

4.1 Types of Graph Two different types of graph are used in our experiments: *full graph* (a short name for fully-connected graph), and *sparse graph*. A graph V with n nodes is constructed based on data points $x_1, \cdots, x_n$, as follows:

1. *Full graph*: Every pair of nodes are connected by an edge, and the edge weight is defined as [24]:

$$A^G_{ij} = \exp\left(-\frac{\|x_i - x_j\|_2^2}{\sigma_i \sigma_j}\right) \tag{4.11}$$

   where the *local scale* parameter σ_i of each data point is set to be the distance between x_i and its $\hat{k}$-th neighbor. We use $\hat{k} = 7$ as suggested in [24], which is shown to be effective in many types of artificial and real-world data sets.

2. *Sparse graph*: Every node is connected to its k_n nearest neighbors. More precisely, Let

$$N(i) = \{j | x_j \text{ is one of the } k_n \text{ nearest neighbors of } x_i, j \neq i\} \tag{4.12}$$

   Define the edge weights $\hat{A}^G_{ij}$ as:

$$\hat{A}^G_{ij} = \begin{cases} A^G_{ij}, \text{ if } i \in N(j) \text{ or } j \in N(i) \\ 0, \text{ otherwise} \end{cases} \tag{4.13}$$

   where A^G is defined as in (4.11). We choose $k_n = \lfloor \log_2 n \rfloor + 1$ as suggested in [20].

Note that the similarity matrix A formed using A^G or $\hat{A}^G$ is usually nonnegative and indefinite.

4.2 Artificial Graph Data To evaluate the algorithms for SymNMF, we first perform experiments with all the Newton-like algorithms proposed in Section 3 – PNewton, $I1$, $I2$, $I1 + I2$ – on a data set used in [24] for graph clustering[3]. The data set consists of

[3] http://webee.technion.ac.il/~lihi/Demos/SelfTuningClustering.html

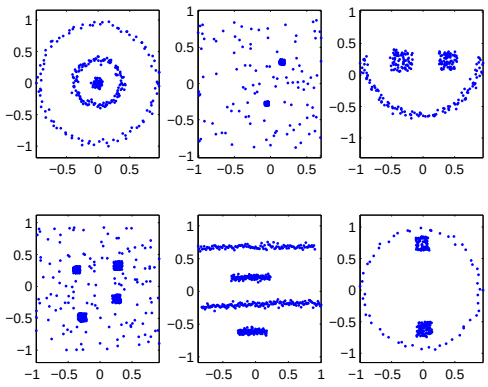


Figure 3: Artificial graph data, with 6 different cases. (reproduced from [24])

Table 5: Computation time on graph data (in seconds). The fastest SymNMF algorithm for each graph is highlighted.

Graph	*1*	*2*	*3*	*4*	*5*	*6*
Spectral	0.09	0.10	0.07	0.61	0.43	0.05
PGD	2.09	0.66	3.17	20.18	24.67	1.28
PNewton	1.68	0.62	0.92	11.32	19.12	0.79
I1	1.18	0.53	0.75	9.57	15.56	0.56
I2	0.68	0.48	0.61	4.77	**4.98**	**0.13**
I1+I2	**0.36**	**0.24**	**0.31**	**3.21**	5.30	0.14

(a) Average computation time on full graph.

Graph	*1*	*2*	*3*	*4*	*5*	*6*
Spectral	0.05	0.10	0.03	0.60	0.14	0.05
PGD	2.45	0.82	2.86	10.25	15.86	1.42
PNewton	7.46	0.50	10.41	26.73	440.87	1.35
I1	2.88	0.48	2.94	17.29	88.80	0.84
I2	7.23	0.37	4.29	2.15	51.88	0.17
I1+I2	**1.71**	**0.26**	**2.07**	**1.75**	**7.94**	**0.14**

(b) Average computation time on sparse graph.

6 artificially constructed cases with data points in 2-dimensional space, shown in Fig. 3.

We compare spectral clustering and SymNMF algorithms on both full and sparse graphs. 20 runs with different initializations are conducted for each of the 6 cases, and the same initializations are used across all SymNMF algorithms for a fair comparison. The average computation time and clustering quality are reported in Tables 5 and 6.

4.2.1 Efficiency From Table 5, we can see that it is essential to combine acceleration techniques in $I1$ and $I2$ in order to achieve an efficient algorithm for SymNMF. Newton-like algorithms, which utilizes second-order information, usually require fewer iterations than PGD. However, a simple algorithm such as standard PNewton, while faster than PGD in many cases, can be much slower as well, as shown in many cases with sparse graph. Among these Newton-like algorithms, only $I1 + I2$ is substantially faster than PGD in every case. We also notice that the current SymNMF algorithms are much slower than spectral clustering that calls an eigensolver and K-means. We will improve the efficiency of SymNMF on similarity matrices with sparse formats in future study.

Table 6: Clustering quality on graph data.

Graph	*1*	*2*	*3*	*4*	*5*	*6*
Spectral	14	18	13	7	14	19
PGD	19	17	18	17	15	20
PNewton	19	19	18	18	16	20
I1	19	19	18	18	16	20
I2	20	20	17	18	16	20
I1+I2	18	18	16	18	17	20

(a) Number of optimal assignments among 20 runs on full graph.

Graph	*1*	*2*	*3*	*4*	*5*	*6*
Spectral	7	16	2	10	1	18
PGD	16	18	16	16	14	20
PNewton	16	20	16	18	14	20
I1	16	20	16	17	14	20
I2	15	20	17	19	11	20
I1+I2	14	17	18	18	11	20

(b) Number of optimal assignments among 20 runs on sparse graph.

4.2.2 Clustering Quality The 6 cases in Fig. 3 have clearly separable cluster structures, which allows an effective method to give completely correct clustering results if initialized properly. Thus, we evaluate the clustering quality of these algorithms by comparing the numbers of optimal clustering assignments. From Table 6, we can see that all the algorithms for SymNMF achieve better clustering quality than spectral clustering on average. However, $I1 + I2$ does not always make improvement over PGD, possibly due to its inexpensive but approximate way to evaluate the reduced Hessian. Another important observation is that SymNMF is not as sensitive as spectral clustering in terms of which graph is used. Although all the algorithms experience performance degradation when we switch from full graph to sparse graph, the impact on SymNMF algorithms is much smaller than the impact on spectral clustering, especially in the cases of Graph 3 & 5.

4.3 Real-world data

4.3.1 Data Sets Document clustering experiments are conducted on two widely-used data sets: TDT2[4] and Reuters-21578[5]. TDT2 is a collection of news articles from various sources (e.g. NYT, CNN, VOA) in 1998. Reuters-21578 is a benchmark text collection from the Reuters newswire in 1987. Both data sets are manually labeled to a set of topics, where each document is assigned to one label in TDT2, and one or more labels in Reuters-21578. Documents with multiple labels are discarded in our experiments. Additionally, clusters representing different topics are highly unbalanced in size, ranging from 1 to thousands of documents. To improve the reliability of our evaluations, we remove any cluster with less than 5 documents.

Image clustering experiments are conducted on a data set for object recognition: COIL-20[6]. COIL-20 contains 128×128 gray-scale images of 20 objects. The shooting directions are equally spaced in the entire 360^o range, resulting in 72 images for each object. The identity information of the objects is used as ground-truth labels.

4.3.2 Experimental Settings In the experiments on real-world data sets, we use $I1 + I2$ as the SymNMF algorithm (due to its consistently better behavior in terms of efficiency and clustering quality), and only construct sparse graphs. Despite the low clustering quality with sparse graphs constructed on the above graph data, sparse graph is generally preferable to full graph in real-world data in terms of clustering quality, as discovered in many previous works [20].

We compare the clustering qualities given by the following 6 algorithms:

1. *Standard K-means and Spherical K-means (SphK-means).* The original data matrix X is used as input to the `kmeans` function in Matlab 7.9 (R2009b). To measure the dissimilarity between two data points x_i, x_j, standard K-means uses $\|x_i - x_j\|_2^2$, and spherical K-means uses $1 - \cos(x_i, x_j)$. The `kmeans` function includes a *batch-update* phase and an additional *online-update* phase in each run. We use both phases on image data. On text data, for efficiency reasons, only the batch-update phase is used.

2. *NMF*: We use the alternating nonnegative least squares algorithm [11] to obtain a solution of the standard form of NMF (1.2). For text data, the data matrix X is normalized such that $\|x_i\|_2 = 1$ for each data point x_i. In addition, X is transformed into its normalized-cut weighted version $XD^{-1/2}$ [22], where D is defined in Section 2.2 with $A^G = X^T X$ (inner-product similarity).

3. *GNMF*: Cai et al. [3] proposed GNMF by adding a graph-theoretic penalty term to (1.2) that takes neighboring relationship into account, so that the resulting method is better at clustering on manifolds. We use the GNMF algorithm and the suggested parameters in [3]. The data matrix X is constructed in the same way as in NMF. However, the neighboring relationship (based on the sparse graph) is generated using the original data matrix.

4. *Spectral clustering*: We use the spectral clustering algorithm proposed by Ng et al. [16]. K-means is used as the final step to obtain clustering results, which is initialized by randomly choosing k samples as centroids.

5. *SymNMF*: The Newton-like algorithm $I1 + I2$ is used.

For each data set, we construct subsets by selecting k groups randomly from the entire data set, and run clustering algorithms on the same subsets for comparisons. 20 random subsets are generated for each k. The algorithms are run 5 times on each subset, with different initializations and the correct cluster number k as input. NMF and GNMF have the same initializations of C and G. Although the data sets are labeled, the labels are only used when evaluating the clustering quality, not by the clustering algorithms.

4.3.3 Results and Analysis We use *clustering accuracy*, the percentage of correctly clustered items given by the maximum bipartite matching, to evaluate the clustering quality (please refer to more details in [22]). The experiment results on TDT2, Reuters-21578, and COIL-20 data sets are shown in Table 7, 8, 9, respectively. We report: 1. The mean of clustering accuracy for each subset, averaged over 20 subsets; 2. The clustering accuracy corresponding to the lowest objective function value on each subset, averaged over 20 subsets. We have the following observations on these results:

1. The overall performance of SymNMF is significantly better than that of the other algorithms. (Note that the clustering accuracy results do not exhibit any trend with increasing k, due to the randomness in selecting subsets from the entire data set.)

[4] `http://projects.ldc.upenn.edu/TDT2/`

[5] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

[6] `http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php`

Table 7: Clustering accuracy on TDT2 data set. The highest clustering accuracy for each k is highlighted.

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=12$	$k=14$	$k=16$	$k=18$	$k=20$
Kmeans	0.8653	0.7994	0.6806	0.6147	0.5805	0.5648	0.5082	0.5286	0.5431	0.5386
SphKmeans	0.8642	0.7978	0.6840	0.6208	0.5777	0.5728	0.5069	0.5305	0.5496	0.5436
NMF	**0.9991**	0.9440	0.8732	0.8292	0.7475	0.7697	0.6910	0.6709	0.7107	0.6695
GNMF	0.9163	0.9150	0.8405	0.8200	0.7344	0.7361	0.6946	0.6812	0.7092	0.6746
Spectral	0.9354	0.9093	0.7719	0.7357	0.6590	0.6308	0.5813	0.5959	0.6346	0.5945
SymNMF	0.9987	**0.9668**	**0.8892**	**0.8819**	**0.8516**	**0.8082**	**0.7874**	**0.7635**	**0.7916**	**0.7686**

(a) The mean of clustering accuracy, averaged over 20 subsets.

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=12$	$k=14$	$k=16$	$k=18$	$k=20$
Kmeans	0.9522	0.8847	0.6916	0.6485	0.5859	0.5993	0.5236	0.5598	0.5620	0.5309
SphKmeans	0.9521	0.8497	0.7141	0.6246	0.5874	0.5998	0.5173	0.5490	0.5646	0.5462
NMF	**0.9991**	0.9469	0.8903	0.8692	0.7639	0.7814	0.6988	0.6951	0.7371	0.6821
GNMF	0.9294	0.9428	0.8929	0.8400	0.7838	0.7497	0.7037	0.6999	0.7484	0.6816
Spectral	0.9983	0.9741	0.8179	0.7557	0.7016	0.6573	0.6145	0.6146	0.6530	0.6199
SymNMF	0.9987	**0.9815**	**0.9119**	**0.9124**	**0.8707**	**0.8085**	**0.8018**	**0.7789**	**0.7895**	**0.7886**

(b) The best of clustering accuracy, averaged over 20 subsets.

Table 8: Clustering accuracy on Reuters-21578 data set. The highest clustering accuracy for each k is highlighted.

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=12$	$k=14$	$k=16$	$k=18$	$k=20$
Kmeans	0.7867	0.5755	0.5137	0.5170	0.4191	0.4377	0.4529	0.3712	0.3403	0.3163
SphKmeans	0.7802	0.5738	0.5197	0.5049	0.4140	0.4383	0.4507	0.3687	0.3421	0.3143
NMF	**0.9257**	0.7737	0.6934	0.6747	0.5568	0.5748	0.5654	0.4608	0.4313	0.4005
GNMF	0.8709	0.7798	**0.7439**	0.6758	**0.7038**	**0.6502**	0.6160	0.5338	0.5704	0.4892
Spectral	0.8885	0.7171	0.6452	0.6452	0.5428	0.5442	0.5637	0.5001	0.4411	0.4338
SymNMF	0.9111	**0.8077**	0.7265	**0.7343**	0.6842	0.6420	**0.6539**	**0.6688**	**0.6188**	**0.6105**

(a) The mean of clustering accuracy, averaged over 20 subsets.

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=12$	$k=14$	$k=16$	$k=18$	$k=20$
Kmeans	0.8750	0.5868	0.4972	0.5196	0.4254	0.4548	0.4766	0.3715	0.3376	0.3186
SphKmeans	0.8945	0.6126	0.5281	0.5351	0.4195	0.4411	0.4770	0.3743	0.3366	0.3154
NMF	**0.9332**	0.7919	0.7087	0.6879	0.5675	0.6021	0.5799	0.4768	0.4406	0.4183
GNMF	0.8915	0.8010	**0.7647**	0.7067	**0.7248**	**0.6784**	0.6260	0.5387	0.5855	0.4814
Spectral	0.8920	0.7467	0.6754	0.6653	0.5813	0.5755	0.5803	0.5337	0.4569	0.4515
SymNMF	0.9116	**0.8248**	0.7387	**0.7493**	0.6820	0.6530	**0.6612**	**0.6723**	**0.6188**	**0.6263**

(b) The best of clustering accuracy, averaged over 20 subsets.

2. By showing the best clustering accuracy selected among 5 runs for each subset, we intend to alleviate the impact of random initializations on the variance of clustering quality in these algorithms, and SymNMF still remains highly competitive. In a few cases where GNMF achieves the highest clustering accuracy, SymNMF has slightly lower clustering accuracy. However, SymNMF can be much better than GNMF in other cases.

3. In Section 1, we attribute the success of NMF on document clustering to the assumption that each basis vector is a good representation of a topic. However, comparing the clustering accuracies of NMF and SymNMF, we can see that this assumption on text corpus is over simplifying. The higher clustering accuracy given by SymNMF implies nonlinear cluster structures in the data. Also, comparing the clustering accuracies of spectral clustering and SymNMF, we can see that the formulation of SymNMF is a more effective way to reveal these nonlinear cluster structures based on neighboring relationship.

4. Spherical K-means uses cosine similarity and was proposed to improve the accuracy of document clustering over standard K-means [6]. These two algorithms have comparable performances in our experiments on text data sets, possibly due to the removal of the online-update phase. However,

Table 9: Clustering accuracy on COIL-20 data set. The highest clustering accuracy for each k is highlighted.

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=20$
Kmeans	0.9206	0.7484	0.7443	0.6541	0.6437	0.6083
SphKmeans	0.8918	0.6906	0.6867	0.6242	0.6154	0.5598
NMF	0.9291	0.7488	0.7402	0.6667	0.6238	0.4765
GNMF	0.9345	0.7325	0.7389	0.6352	0.6041	0.4638
Spectral	0.7925	0.8115	0.8023	0.7969	0.7372	0.7014
SymNMF	**0.9917**	**0.8406**	**0.8725**	**0.8221**	**0.8018**	**0.7397**

(a) The mean of clustering accuracy, averaged over 20 subsets. (The last column shows the results after 20 runs of each algorithm on the entire data set.)

	$k=2$	$k=4$	$k=6$	$k=8$	$k=10$	$k=20$
Kmeans	0.9330	0.7977	0.8263	0.7158	0.7037	0.6840
SphKmeans	0.8941	0.7833	0.7464	0.6926	0.6756	0.5694
NMF	0.9288	0.7467	0.7203	0.6612	0.6233	0.4674
GNMF	0.9323	0.7495	0.7413	0.6602	0.6210	0.4688
Spectral	**0.9924**	0.8990	0.9197	**0.8916**	0.8171	0.7479
SymNMF	0.9917	**0.9097**	**0.9557**	0.8863	**0.8597**	**0.7972**

(b) The best of clustering accuracy, averaged over 20 subsets. (The last column shows the results after 20 runs of each algorithm on the entire data set.)

on COIL-20 image data set, spherical K-means is consistently worse than standard K-means, which means that measuring cosine similarity is especially inappropriate when different clusters cannot be represented by different basis vectors.

5 Related Work

We have developed an algorithm for SymNMF with guaranteed convergence to a stationary point in this paper. Our Newton-like algorithm, to the best of our knowledge, is different from all previous algorithms for factorizing a symmetric matrix. As mentioned in Section 1, the nonnegative factorization of a kernel matrix (1.3) appeared in Ding et al. [7], where the kernel matrix K is positive semi-definite *and* nonnegative. Their algorithm for solving (1.3) is a variation of the multiplicative update rule algorithm for NMF [4]. When K has negative entries, this algorithm for symmetric NMF is not applicable because it introduces negative entries into H and violates the constraint $H \geq 0$. Later, Li et al. [14] proposed another multiplicative update rule algorithm for their formulation of factorizing the kernel matrix K in semi-supervised clustering, where K is assumed to be positive semi-definite but not necessarily nonnegative. However, it was pointed out in Gonzales and Zhang [8] that multiplicative update rule algorithms do not have the property that every limit point is a stationary point. We tested both algorithms in [7] and [14] on the graph clustering example shown in Fig. 1a, by replacing K with the similarity matrix A which is indefinite. Both algorithms failed to converge to a stationary point.

6 Conclusion

In this paper, we proposed Symmetric NMF (SymNMF) as a new framework of graph clustering methods. A graph is represented in a matrix A containing pairwise similarity values, and SymNMF performs a nonnegative symmetric factorization of A. The formulation of SymNMF is $\min_{H \geq 0} \|A - HH^T\|_F^2$, which can be related to a generalized form of many graph clustering objectives.

Compared to spectral clustering, the nonnegativity constraint in SymNMF enables H to be easily interpreted into clustering assignments and to naturally capture the cluster structure embedded in the graph representation. Compared to NMF, SymNMF allows incorporating similarity between the data points in the problem formulation, and we can employ the most suitable measure to describe the inherent cluster structure.

Our investigation into the limitations of NMF also reveals that it has similar assumptions on the cluster structure with spherical K-means: Both methods attempt to use different basis vectors to represent different clusters respectively. This seems to explain why spherical K-means is often used as an initialization strategy for NMF [21].

We developed an efficient Newton-like algorithm for SymNMF, and showed that SymNMF often achieves higher clustering quality than spectral clustering and NMF in experiments with artificial and real-world data. In addition, choosing the parameters in constructing similarity matrix is a crucial issue in graph clustering, and the clustering results of SymNMF are not as sensitive as spectral clustering to these parameters.

Although the symmetric factorization of a similarity

matrix has been used to build theoretical connection between different clustering methods [7], SymNMF was rarely applied as an independent clustering method in practice. Algorithms with good convergence property, and also comparisons to other clustering methods (e.g. NMF, spectral clustering), were even more rare. In this paper, we strived to develop a *general* framework based on SymNMF, one with minimal constraints and flexible enough for future extensions and applications, such as semi-supervised learning, image segmentation, and recommender systems. The proposed algorithm can easily be parallelized, for example, the evaluation and Cholesky factorization of different diagonal blocks of the Hessian can run in parallel. Since a sparse similarity matrix A is frequently used in clustering, designing sparse algorithms for SymNMF is also an important future topic.

Acknowledgments

The authors would like to thank Sangwoon Yun of Korea Institute for Advanced Study, for his help in designing a coordinate gradient descent algorithm for SymNMF [18], and the anonymous reviewers for their valuable and careful comments.

References

[1] D. P. Bertsekas, "Projected newton methods for optimization problems with simple constraints," *SIAM J. Control and Optimization*, vol. 20, no. 2, pp. 221–246, 1982.

[2] ——, *Nonlinear programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

[3] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," pp. 1548–1560, 2011.

[4] M. Catral, L. Han, M. Neumann, and R. J. Plemmons, "On reduced rank nonnegative matrix factorization for symmetric matrices," *Linear Algebra and Its Applications*, vol. 393, pp. 107–126, 2004.

[5] I. Dhillon, Y. Guan, and B. Kulis, "A unified view of kernel k-means, spectral clustering and graph cuts," University of Texas at Austin, Tech. Rep. TR-04-25, 2005.

[6] I. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, pp. 143–175, 2001.

[7] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *SDM '05: Proc. of SIAM Int. Conf. on Data Mining*, 2005, pp. 606–610.

[8] E. F. Gonzales and Y. Zhang, "Accelerating the lee-seung algorithm for non-negative matrix factorization," Rice University, Tech. Rep. TR05-02, 2005.

[9] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*. Philadelphia, PA: SIAM, 1995.

[10] H. Kim and H. Park, "Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis," *Bioinformatics*, vol. 23, no. 12, pp. 1495–1502, 2007.

[11] ——, "Nonnegative matrix factorization based on alternating non-negativity-constrained least squares and the active set method," *SIAM J. on Matrix Analysis and Applications*, vol. 30, no. 2, pp. 713–730, 2008.

[12] J. Kim and H. Park, "Sparse nonnegative matrix factorization for clustering," Georgia Inst. of Technology, Tech. Rep. GT-CSE-08-01, 2008.

[13] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, "Semi-supervised graph clustering: a kernel approach," in *ICML '05: Proc. of the 22nd Int. Conf. on Machine learning*, 2005, pp. 457–464.

[14] T. Li, C. Ding, and M. I. Jordan, "Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization," in *ICDM '07: Proc. of the 7th IEEE Int. Conf. on Data Mining*, 2007, pp. 577–582.

[15] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[16] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, 2001, pp. 849–856.

[17] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[18] P. Tseng and S. Yun, "A coordinate gradient descent method for nonsmooth separable minimization," *Mathematical Programming*, vol. 117, no. 1, pp. 387–423, 2009.

[19] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[20] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[21] S. Wild, J. Curry, and A. Dougherty, "Improving non-negative matrix factorizations through structured initialization," *Pattern Recognition*, vol. 37, pp. 2217–2232, 2004.

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

[23] S. Yun, P. Tseng, and K.-C. Toh, "A block coordinate gradient descent method for regularized convex separable optimization and covariance selection," *Mathematical Programming*, vol. 129, pp. 331–355, 2011.

[24] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems 17*, 2004, pp. 1601–1608.