Nonnegative Matrix Factorization for Combinatorial Optimization: Spectral Clustering, Graph Matching, and Clique Finding

Chris Ding
CSE Department
University of Texas at Arlington
Arlington, TX 76019
chqding@uta.edu

Tao Li
School of Computer Science
Florida International University
Miami, FL 33199
taoli@cs.fiu.edu

Michael I. Jordan
Dept. of EECS and Dept. of Statistics
University of California at Berkeley
Berkeley, CA 9472
jordan@cs.berkeley.edu

Abstract

Nonnegative matrix factorization (NMF) is a versatile model for data clustering. In this paper, we propose several NMF inspired algorithms to solve different data mining problems. They include (1) multi-way normalized cut spectral clustering, (2) graph matching of both undirected and directed graphs, and (3) maximal clique finding on both graphs and bipartite graphs. Key features of these algorithms are (a) they are extremely simple to implement; and (b) they are provably convergent. We conduct experiments to demonstrate the effectiveness of these new algorithms. We also derive a new spectral bound for the size of maximal edge bicliques as a byproduct of our approach.

keywords: *Nonnegative matrix factorization, clustering, graph matching, clique finding*

1 Introduction

A large number of data mining tasks can be formulated as optimization problems. Examples include K-means clustering, support vector machines (SVM), linear discriminant analysis (LDA), and semi-supervised clustering. Nonnegative matrix factorization (NMF) has been shown to be able to solve several data mining problems including classification and clustering. In this paper, we show that NMF provides an optimization framework which has a much broader applicability and can solve a variety of data mining problems.

NMF has been a significant success story in the machine learning literature. Originally proposed as a method for finding matrix factors with parts-of-whole interpretations [17], NMF has been shown to be useful in a variety of applied settings, including environmetrics [26], chemometrics [34], pattern recognition [20], multimedia data analysis [5], text mining [35, 27], and DNA gene expression analysis [3]. Algorithmic extensions of NMF have been developed to accommodate a variety of objective functions [6, 10, 21] and a variety of data analysis problems, including classification [29] and collaborative filtering [31]. A number of studies have focused on further developing computational methodologies for NMF [15, 1, 22, 36, 23]. Researchers have also begun to explore some of the relationships between matrix factorizations and clustering [11, 9, 10], opening up a variety of additional potential applications for NMF techniques. It has been shown that NMF with the sum of squared error cost function is equivalent to a relaxed K-means clustering [37, 8], the most widely used unsupervised learning algorithm. In addition, NMF with the I-divergence cost function is equivalent to probabilistic latent semantic indexing, another unsupervised learning method popularly used in text analysis.

In this paper, we show that NMF provides a nice framework for solving many data mining optimization problems. In particular, we broaden the scope of application to include several different data mining problems. We provide NMF-inspired solutions to

- Multi-way normalized cut spectral clustering,
- Graph matching,

1550-4786/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDM.2008.130

- Maximal clique and biclique

We also show that new analytical insights flow from this approach. In particular, in the maximal biclique problem our approach allows us to derive a new spectral bound for the size of the maximal edge clique.

2 Spectral clustering

Normalized Cuts [30] is a NP-hard optimization problem. We focus on the multi-way version of Normalized Cuts, which can be formulated as the minimization of the following objective function:

$$J_{nc} = \sum_{1 \le p < q \le K} \frac{s(C_p, C_q)}{\rho(C_p)} + \frac{s(C_p, C_q)}{\rho(C_q)} \quad (1)$$

where w_{ij} are the entries of an affinity matrix W, $\{C_i\}$ are disjoint subsets of the vertices, $d_i = \sum_j w_{ij}$, $\rho(C_k) = \sum_{i \in C_k} d_i$, and $s(C_k, C_\ell) = \sum_{i \in C_k} \sum_{j \in C_\ell} w_{ij}$. Let $\mathbf{h}_k = \{0,1\}^n$ be an indicator vector for cluster C_k. Let $D = \text{diag}(d_1, \cdots, d_n)$. We have

$$J_{nc} = \sum_{\ell=1}^{K} \frac{\mathbf{h}_\ell^T (D-W) \mathbf{h}_\ell}{\mathbf{h}_\ell^T D \mathbf{h}_\ell} = K - \sum_{\ell=1}^{K} \frac{\mathbf{h}_\ell^T W \mathbf{h}_\ell}{\mathbf{h}_\ell^T D \mathbf{h}_\ell}.$$

Let $H = (\mathbf{h}_1/||D^{1/2}\mathbf{h}_1||, \cdots, \mathbf{h}_K/||D^{1/2}\mathbf{h}_K||)$. The problem becomes [13]

$$\max_{H^T D H = I,\ H \ge 0} \text{Tr}(H^T W H). \quad (2)$$

If we ignore the nonnegative constraints, and keep the orthogonality intact, the solution for H is given by the generalized eigenvectors of $D - W$. However, the mixed signs of the eigenvector solutions make the cluster assignment difficult. Thus the nonnegativity constraints is the key.

2.1 NMF algorithm

Lee and Seung [19] showed that the NMF problem could be solved by a multiplicative update algorithm. In this section we show that a similar approach can be adopted for Normalized Cuts. We propose the following multiplicative update algorithm for solving Eq. (2):

$$H_{ij} \leftarrow H_{ij} \sqrt{\frac{(WH)_{ij}}{(DH\alpha)_{ij}}},\ \alpha \equiv H^T W H. \quad (3)$$

In the following two subsections we show that this update yields a correct solution at convergence and we show that the algorithm is guaranteed to converge.

2.1.1 Correctness

Theorem 1 *Fixed points of Eq. (3) satisfy the KKT condition.*

Proof. We begin with the Lagrangian

$$L = \text{Tr}\, H^T W H - \text{Tr}\, \alpha (H^T D H - I), \quad (4)$$

where the Lagrange multiplier α enforces the orthogonality condition $H^T D H = I$. The nonnegativity constraint is enforced using the KKT complementary slackness condition

$$(WH - DH\alpha)_{ij} H_{ij} = 0. \quad (5)$$

Summing over j, we obtain $(H^T W H)_{ii} = (H^T D H \alpha)_{ii} = \alpha_{ii}$. This gives the diagonal elements of α. To find the off-diagonal elements of α, we temporally ignore the nonnegativity requirement. This gives $(WH - DH\alpha)_{ij} = 0$. Left-multiplying by $H_{i'j}$ and summing over j, we obtain $(H^T W H)_{i'i} = \alpha_{i'i}$ for the off-diagonal elements of α. Combining these two results yields

$$\alpha = H^T W H. \quad (6)$$

Clearly, a fixed point of the update rule Eq. (3) satisfies $(WH - DH\alpha)_{ij} H_{ij}^2 = 0$, which is identical to Eq. (5). This is so because if $H_{ij} = 0$, we have $H_{ij}^2 = 0$, and vice versa. □

2.1.2 Convergence

Theorem 2 *Under the update rule of Eq. (3), the Lagrangian function L of Eq. (4) increases monotonically (it is nondecreasing).*

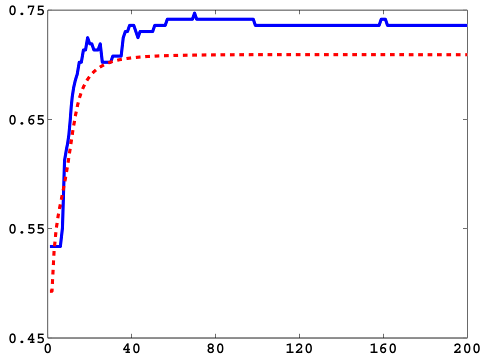
The proof is given in the Appendix.

2.2 Initialization

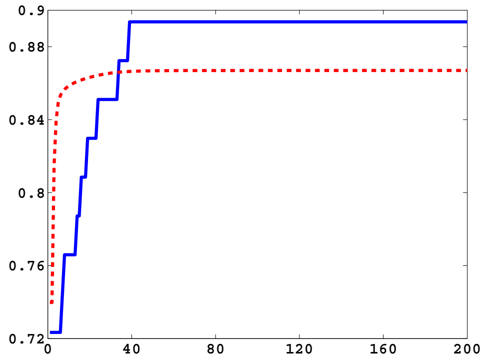
An approximate solution to the normalized cut problem can be obtained by K-means clustering in an eigenspace embedding [25]. We use such a clustering to initialize H. Specifically, we (1) compute the K principal eigenvectors of $D^{-1/2} W D^{-1/2}$, (2) normalize each embedded point to the unit sphere, and (3) perform K-means on the normalized points. This gives H_0. We then initialize the update rule Eq. (3) with $H_0 + 0.2$ and iterate until convergence.

2.3 Experiments

We report results from running the NMF algorithm on the datasets *wine* and *soybean* from the UCI data repository. Figure 1(a) and Figure 1(b) plot the clustering accuracy



(a) Wine Dataset



(b) Soybean Dataset

Figure 1. Objective function (dashed curve) and clustering accuracy (solid curve) as a function of iterations on wine and soybean datasets. On wine dataset, objective function is scaled by a factor of 0.33 to be comparable to clustering accuracy values. On soybean dataset, objective function is scaled by a factor of 0.22.

across iterations of the algorithm; we also plot the evolution of the objective function L. Note that the algorithm yields a significant improvement relative to its initialization from the spectral clustering algorithm of [25]. Note also that (as expected from Theorem 2), the objective function increases monotonically.

3 Graph matching

Graph matching plays an important role in many data mining applications such as pattern recognition, information retrieval, and frequent pattern mining to determine correspondences between the components (vertices and edges) of two attributed structures [16]. Given two graphs with adjacency matrices A and B, the graph matching problem is to permute the nodes of A such that $\sum_{ij}[(P^TAP)_{ij} - B_{ij}]^2 = ||P^TAP - B||^2$ is minimized, where P is a permutation matrix. Since $||P^TAP - B||^2 = ||A|| + ||B|| - 2\text{Tr}\, P^TAPB^T$, the problem becomes

$$\max_{P^TP=I, P\geq 0} \text{Tr}\, P^TAPB^T, \tag{7}$$

The constraints $P^TP = PP^T = I$ and $P \geq 0$ together ensure that each row has one nonzero elements, and so does each column. We first consider undirected graphs, i.e., $A = A^T, B = B^T$.

3.1 NMF algorithm for undirected graph matching

In this section we show that the following multiplicative update algorithm:

$$P_{ij} \leftarrow P_{ij}\sqrt{\frac{(APB)_{ij}}{(P\alpha)_{ij}}}, \quad \alpha \equiv \frac{P^TAPB + (P^TAPB)^T}{2}, \tag{8}$$

is correct and converges to a locally optimal solution. This algorithm has $O(n^3)$ complexity. At convergence, the elements of the solution P^∞ are not generally $\{0, 1\}$-valued and rounding is necessary. We propose to carry out this rounding via the Hungarian algorithm for the bipartite graph matching. This algorithm, which can be formulated as $P = \arg\max_P \text{Tr}(PP^\infty)$, also has complexity $O(n^3)$.

3.1.1 Correctness

Theorem 3 *Fixed points of Eq. (8) satisfy the KKT condition.*

Proof. The Lagrangian function is

$$L(P) = \text{Tr}\, P^TAPB - \text{Tr}\, \alpha(P^TP - I). \tag{9}$$

The KKT complementary condition for the nonnegativity constraint is

$$(APB - P\alpha)_{ij}P_{ij} = 0. \tag{10}$$

Summing over j, we obtain $(P^TAPB)_{ii} = (P^TP\alpha)_{ii} = \alpha_{ii}$. This gives the diagonal elements of α. To find the off-diagonal elements of α, we temporally ignore the nonnegativity requirement and setting $\partial L/\partial P = 0$, we obtain $(P^TAPB)_{ij} = \alpha_{ij}$ for the off-diagonal elements of α. Combining these two results together, we have $\alpha = P^TAPB$. However, since $P^TP - I$ is symmetric,

$$\begin{aligned}\text{Tr}\,\alpha(P^TP-I) &= \text{Tr}\,[\alpha(P^TP-I)]^T \\ &= \text{Tr}\,(P^TP-I)\alpha^T = \text{Tr}\,\alpha^T(P^TP-I)\end{aligned}$$

Thus only the symmetric part of α contributes to L, i.e., α should also be symmetric. Thus we set

$$\alpha = [P^TAPB + (P^TAPB)^T]/2. \tag{11}$$

Clearly, the fixed points of the update rule Eq. (8) satisfy $(APB - P\alpha)_{ij}P_{ij}^2 = 0$, which is identical to Eq. (10). □

3.1.2 Convergence

Theorem 4 *Under the update rule of Eq. (8), the Lagrangian function $L(P)$ of Eq. (9) increases monotonically (nondecreasing).*

The proof is similar to that of Theorem 1 and is omitted.

3.1.3 Initialization

Let $J = P^TAPB^T$. It can be shown that J has an upper bound [33] and we can solve the optimization problem based on this upper bound to initialize our algorithm. Let the eigen-decompositions of A, B be $A = U\Sigma U^T$ and $B = V\Lambda V^T$, where $\Sigma = \text{diag}(\sigma_1, \cdots, \sigma_n)$ contains the eigenvalues of A in descending order, and $\Lambda = \text{diag}(\lambda_1, \cdots, \lambda_n)$ contains the eigenvalues of B in descending order. We have

$$J = \text{Tr}\,P^T(U\Sigma U^T)P(V\Lambda V^T) = \text{Tr}\,\Sigma(U^TPV)\Lambda(U^TPV)^T$$

Thus an upper bound of J is attained

$$\max_{P^TP=I,P\geq 0} \text{Tr}\,P^TAPB \leq \max_{P^TP=I} \text{Tr}\,P^TAPB = \text{Tr}\,\Sigma\Lambda, \tag{12}$$

by relaxing P from a permutation matrix to an orthonormal matrix: $P = UV^T$. However, the solution is not unique, because in the decomposition $A = U\Sigma U^T$, the signs of the eigenvectors are not unique. That is, $A = U\Sigma U^T$ holds for any combination of signs of all eigenvectors: $U = (\pm\mathbf{u}_1, \cdots, \pm\mathbf{u}_n)$.

To resolve this non-uniqueness, we adopt Umeyama's approach [32]. Let $\bar{U}$ be the matrix containing the absolute values of U: $(\bar{U})_{ij} = |U_{ij}|$, and similarly for $\bar{V}$. We compute

$$P_0 = \bar{U}\bar{V}^T. \tag{13}$$

It is easy to see that $0 \leq (P_0)_{ij} \leq 1$, because each row of $\bar{U}$ and $\bar{V}$ is a nonnegative vector with length 1. We use P_0 as the initial value for our iterative updating algorithm.

3.2 NMF algorithm for directed graph matching

Graph matching for directed graphs is harder than for undirected graphs and there has been little research on this topic. It is thus of some interest that our approach to graph matching generalizes to directed graphs in a straightforward way. In this case $A \neq A^T, B \neq B^T$. The updating rule that we propose is

$$P_{ij} \leftarrow P_{ij}\sqrt{\frac{(APB^T + A^TPB)_{ij}}{(2P\alpha)_{ij}}}, \tag{14}$$

where

$$\alpha = \frac{P^T(APB^T + A^TPB) + (APB^T + A^TPB)^TP}{4}. \tag{15}$$

We can readily establish correctness and convergence for this update rule using arguments similar to those for the undirected case.

The initialization of the algorithm is somewhat more delicate than for the undirected case. Indeed, for directed graphs, no upper bound is known to exist. Following [32], we base our initialization on the following two hermitian matrices:

$$\tilde{A} = \frac{A + A^T}{2} + i\frac{A - A^T}{2},\ \widetilde{B} = \frac{B + B^T}{2} + i\frac{B - B^T}{2}$$

where $i = \sqrt{-1}$. We can obtain an upper bound for $max_P||P^T\tilde{A}P - \widetilde{B}||^2$ using the same argument as in Section 4.1.3. Let the eigendecompositions be $\tilde{A} = U\Sigma U^T$, $\widetilde{B} = V\Sigma V^T$. Note that U, V are complex valued matrices. Let $\overline{U}$ contains the magnitudes of each element of U $\overline{U}_{ij} = |U_{ij}|$, and similarly for $\overline{V}$. Then $P_0 = \overline{U}\,\overline{V}^T$ gives the initial value of P. When $A = A^T, B = B^T$, this algorithm reduces to that for undirected graphs.

3.3 Experiments

Following standard procedures in the literature on matching algorithms, we test our algorithm by considering matrices $A_{ij} = 100r_{ij}$ where r_{ij} is a uniform random number in $[0, 1]$ and $B_{ij} = (P_t^TAP_t)_{ij}(1 + \epsilon r'_{ij})$, where P_t is a permutation and ϵ sets the noise level. Note that the globally optimal permutation P satisfies $||P^TAP - B||^2 \leq ||P_t^TAP_t - B||^2$. Because it is difficult to compute the true global solution in experiments, we consider a solution to be optimal if it satisfies this inequality.

The state-of-art practical algorithm for directed graph-matching is Umeyama's algorithm [32]. This algorithm computes $P = \arg\max_P \mathrm{Tr}\ PP_0$ using the Hungarian algorithm. We compare to Umeyama's algorithm. We first present an example with $N = 6$ and $\epsilon = 0.4$. The input weighted graphs (matrices) are

$$P^TAP = \begin{pmatrix} 0 & 32 & 46 & 48 & 16 & 51 \\ 32 & 0 & 29 & 51 & 36 & 67 \\ 46 & 29 & 0 & 26 & 52 & 52 \\ 48 & 51 & 26 & 0 & 32 & 68 \\ 16 & 36 & 52 & 32 & 0 & 84 \\ 51 & 67 & 52 & 68 & 84 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 & 20 & 30 & 47 & 42 & 58 \\ 20 & 0 & 24 & 64 & 34 & 76 \\ 30 & 24 & 0 & 31 & 70 & 42 \\ 47 & 64 & 31 & 0 & 23 & 71 \\ 42 & 34 & 70 & 23 & 0 & 82 \\ 58 & 76 & 42 & 71 & 82 & 0 \end{pmatrix}$$

where for ease of inspection, we permute A back to best match B using the solution P. P_0 and P^∞ at convergence are shown below:

$$P_0 = \begin{pmatrix} 0.76 & 0.55 & 0.79 & 0.73 & 0.85 & \underline{0.98} \\ 0.76 & 0.67 & 0.77 & \underline{0.89} & 0.61 & 0.61 \\ 0.73 & 0.55 & 0.83 & 0.79 & \underline{0.90} & 0.94 \\ 0.89 & 0.72 & \underline{0.99} & 0.93 & 0.63 & 0.91 \\ \underline{0.98} & 0.83 & 0.88 & 0.88 & 0.46 & 0.82 \\ 0.87 & \underline{0.97} & 0.69 & 0.62 & 0.50 & 0.67 \end{pmatrix}$$

$$P^\infty = \begin{pmatrix} 0.38 & 0.07 & 0.00 & 0.00 & \underline{0.27} & 0.14 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.35 & \underline{0.75} \\ 0.00 & 0.00 & 0.00 & \underline{0.82} & 0.24 & 0.02 \\ 0.00 & 0.00 & \underline{0.91} & 0.07 & 0.03 & 0.00 \\ \underline{0.78} & 0.00 & 0.12 & 0.00 & 0.00 & 0.08 \\ 0.00 & \underline{0.97} & 0.00 & 0.06 & 0.01 & 0.00 \end{pmatrix}$$

where the underline indicates that the corresponding element is rounded to one and the remaining elements are rounded to zero by the Hungarian algorithm. For this example, our algorithm correctly computes the permutation : $||P^TAP - B|| = 62.42$. Umeyama's algorithm failed to recover the correct permutation: $||P_0^TAP_0 - B|| = 97.96$, but did improve the matching (without the permutation, $||A - B|| = 161.98$).

We run Umeyama and our algorithm on graphs with sizes up to 50 at noise levels up to $\epsilon = 0.4$. The results are shown in Table 2, which presents the success rate of correctly computing the global permutation, averaging over 100 runs for $N = 10$, 50 runs for $N = 50$ and 20 runs for $N = 50$. P_0 gives the results for Umeyama algorithm and NMF for our algorithm. Our algorithm does significantly better than the Umeyama algorithm at higher noise level.

N	ϵ	P$_0$	NMF
10	0.1	0.72	0.97
10	0.2	0.19	0.69
10	0.3	0.04	0.36
10	0.4	0.00	0.19
20	0.2	0	0.74

Table 1. Success rate for different sizes (N) and noise levels (ϵ). P_0: using P_0 and the Hungarian algorithm. NMF: using NMF and the Hungarian algorithm.

N	ϵ	P_0	NMF	P_0 + refine	NMF+refine
50	0.2	0	0.70	0.20	0.95

Table 2. Success rate at $N = 50$. P_0 + refine: using P_0 and refinement. NMF+refine: using NMF and refinement.

We can improve on the basic algorithm by adding an refinement algorithm, which exchanges two or three nodes at a time in a greedy fashion to optimize the matching. Table 2 also presents results for this refinement. We see that refinement can significantly improve both of the basic algorithms.

4 Maximal clique

We base our approach to computing maximal cliques on a theorem due to [24] that relates maximal cliques to the optimization of a quadratic function. Let A be the adjacency matrix of an undirected graph with weights in $\{0, 1\}$. The computation of maximal cliques can be formulated as the solution to the following optimization problem:

$$\max_{\mathbf{x}} \ \mathbf{x}^TA\mathbf{x}, \quad s.t. \sum_{i=1}^n x_i = 1, \ x_i \geq 0, \qquad (16)$$

where $A_{ii} = 0$.

Theorem 5 (Motzkin and Straus) *Let G be an unweighted graph and let $\mathbf{x}^*$ the optimal solution for the problem of Eq. (16). Let $C = \{i \mid x_i^* > 0\}$ be the subset corresponding to nonzero elements. If nonzero elements have the same values, $x_i^* = 1/|C|$ for all $i \in C$ (in which case $\mathbf{x}^*$ is called a characteristic vector of a subset C), C is a maximal clique in G.*

An algorithm for computing solutions to the maximal clique problem has been presented by [28] and [12]. The algorithm has a multiplicative form:

$$x_i^{(t+1)} = x_i^{(t)} \frac{(S\mathbf{x}^{(t)})_i}{[\mathbf{x}^{(t)}]^T S\mathbf{x}^{(t)}} \tag{17}$$

and has been shown to be effective in practice [28, 12].

This approach is interesting not only because it anticipates our work in using a multiplicative update to solve a combinatorial optimization problem, but also because the algorithm provides a very clear link between sparsity and the L_1-norm constraint.

In this section, we use our NMF-based approach to: (1) Prove the convergence of the update algorithm of Eq. (17); (2) generalize the L_1-norm constraint to L_p-norm constraint and derive an algorithm to compute it; (3) generalize this methodology to bipartite graphs.

4.1 L_p-norm constraints

We generalize the maximal clique problem to the following optimization problem:

$$\max_{\mathbf{x}} \mathbf{x}^T A\mathbf{x}, \quad s.t. \sum_{i=1}^{n} x_i^\beta = 1, \; x_i \geq 0, \tag{18}$$

where $\beta \in [1, 2]$ is a parameter. We show the following:

(1) The maximal clique is obtained when $\beta = 1 + \epsilon, \; 0 < \epsilon \ll 1$, while setting $A_{ii} = 1$. As we will see later, setting $A_{ii} = 1$ enables us to generalize this approach to bipartite graphs.

(2) A convergent algorithm can be obtained for $\beta \in [1, 2]$. When $\beta = 1$, this algorithm reduces to the extant algorithm of Eq. (17).

(3) As $\beta \to 1_+$, the sparsity of the solution increases steadily, reflecting the close relation between L_1 constraints and sparsity. At $\beta = 2$, the solution is given by the principal eigenvector of A.

Let $\tilde{A}$ denote the adjacency matrix of the graph with $\tilde{A}_{ii} = 1$. and let A be the adjacency matrix of the graph with $A_{ii} = 0$. We can prove the following Generalized Motzkin-Straus Theorem:

Theorem 6 *Using $\tilde{A}$ as the adjacency matrix, and setting $\beta = 1 + \epsilon, \; 0 < \epsilon \ll 1$. Let $C = \{i \mid x_i > 0\}$ be the subset corresponding to nonzero elements. If nonzero elements have same values, $x_i = 1/|C|$ for $\forall i \in C$ C is a maximal clique in G.*

Proof sketch. Since the nonzero elements of $\mathbf{x}$ have constant values, $\mathbf{x}^*$ must have the form $\mathbf{x}^* = (1/|C|^{1/\beta})(1 \cdots 1, 0 \cdots 0)^T$, if we index the nodes of C first. The objective becomes $J = (\mathbf{x})^T \tilde{A}\mathbf{x} = |C|^{2-2/\beta}$. Because $2 - 2/\beta > 0$, $\max J$ is equivalent to $\max |C|$. □

Note that if we use $\beta = 1$, then J is equal to one independent of $|C|$; i.e., we are not guaranteed to compute the maximal clique. Theorem 6 can be readily generalized to bipartite graphs.

4.1.1 An algorithm for the generalized Motzkin-Strauss theorem

When $\tilde{A}$ is positive definite, there is a unique *global* solution to the quadratic optimization problem. When $\tilde{A}$ is positive semidefinite, there are several solutions with the same optimal value of the objective function. When $\tilde{A}$ is indefinite, there are many locally optimal solutions. In our case, since $A_{ii} = 0$, A is always indefinite. Thus working with the L_p-norm version (Theorem 6) has some advantages.

To find the local maxima and thus the maximal cliques, we use an algorithm that iteratively updates a current solution vector $\mathbf{x}^{(t)}$ as follows:

$$[x_i^{(t+1)}]^\beta = x_i^{(t)} \frac{(S\mathbf{x}^{(t)})_i}{[\mathbf{x}^{(t)}]^T S\mathbf{x}^{(t)}}. \tag{19}$$

The iteration starts with an initial guess $\mathbf{x}^{(0)}$ and is repeated until convergence.

We analyze the basic properties of this algorithm.

Feasibility. First, we show that from any initial $\mathbf{x}^{(0)}$, the iteration will lead to a feasible solution:

$$\sum_i \left[x_i^{(t+1)}\right]^\beta = \sum_i \frac{x_i^{(t)}(S\mathbf{x}^{(t)})_i}{[\mathbf{x}^{(t)}]^T S\mathbf{x}^{(t)}} = 1.$$

Optimality. Second, we show that the update rule satisfies the first-order KKT optimality condition. We form the Lagrangian function

$$L = \mathbf{x}^T S\mathbf{x} - \lambda(\sum_i x_i^\beta - 1) \tag{20}$$

where λ is the Lagrangian multiplier for enforcing the L_p constraint. This leads to the following KKT condition:

$$\left[2(S\mathbf{x})_i - \lambda\beta[x_i]^{\beta-1}\right] x_i = 0. \tag{21}$$

Summing over index i, we obtain the value for the Lagrangian multiplier λ:

$$2\mathbf{x}^T S\mathbf{x} = \lambda\beta \sum_{i=1}^{n} [x_i]^\beta = \lambda\beta. \tag{22}$$

Clearly the updating rule

$$[x_i^{(t+1)}]^\beta = x_i^{(t)} \frac{(S\mathbf{x}^{(t)})_i}{(\lambda\beta/2)}, \tag{23}$$

satisfies the KKT condition Eq. (36) at convergence. Substituting the value of λ from Eq. (22), this yields the update rule Eq. (19).

Convergence.

Theorem 7 *Under the update rule in Eq. (23), the Lagrangian function L of Eq. (20) is monotonically increasing (nondecreasing),*

Since J is bounded from above, the convergence of this algorithm is thus established.

Upper bound.

From the proof of Theorem 6, we can provide a new derivation of a known upper bound on the size of the maximal clique [cf. 2]:

$$|C| = \frac{(\mathbf{x}^*)^T \tilde{A} \mathbf{x}^*}{(\mathbf{x}^*)^T \mathbf{x}^*} \leq \max_{\mathbf{x}} \frac{\mathbf{x}^T \tilde{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} = \lambda_1(\tilde{A}).$$

4.2 Generalize to bipartite graph

Let $G(B)$ be a bipartite graph with a set R of r-nodes and a set C of c-nodes. Let B be the adjacency matrix of $G(B)$. B is a rectangular matrix of $n = |R|$ rows (r-nodes) and $m = |C|$ columns (c-nodes). A biclique is a subset (R_1, C_1), where $R_1 \subset R$ and $C_1 \subset C$, such that every r-node in R_1 is connected to every c-node in C_1. There are two types of maximal bicliques: (a) maximal edge bicliques where the number of edges, $|R_1| \cdot |C_1|$, is maximal and (b) maximal node bicliques where the number of nodes, $|R_1| + |C_1|$, is maximal. Typically, maximal edge biclique selects the largest block area in the adjacency matrix and is the interesting biclique. Maximum-node biclique is typically a narrow/skinny block in the adjacency matrix and is not very useful.

A maximal biclique is computed via the solution to the following optimization problem:

$$\max_{\mathbf{x} \in F_x^\alpha,\, \mathbf{y} \in F_y^\beta} \mathbf{x}^T B \mathbf{y} \tag{24}$$

where the region F_x^α is defined as $F_x^\alpha : \sum_{j=1}^m x_i^\alpha = 1,\ x_i \geq 0$. The region F_y^β can be defined similarly.

Let $(\mathbf{x}^*, \mathbf{y}^*)$ be an optimal solution, let $R_1 = \{i \mid x_i^* > 0\}$ be the subset of nonzero elements in $\mathbf{x}^*$, and let $C_1 = \{j \mid y_j^* > 0\}$ be the subset of nonzero elements in $\mathbf{y}^*$. We can derive a Generalized Motzkin-Straus Theorem for bipartite graphs:

Theorem 8 *Let $\beta = 1 + \epsilon, 0 < \epsilon \ll 1$. For an optimal solution $(\mathbf{x}^*, \mathbf{y}^*)$, if nonzero elements of $\mathbf{x}^*$ have the same values, and if nonzero elements of $\mathbf{y}^*$ have the same values, then (R_1, C_1) is a maximal edge biclique in B. The objective function has the optimal value $J = (|R_1||C_1|)^{1-1/\beta}$.*

We provide an iterative algorithm to compute the maximal edge biclique. We prove its feasibility, correctness, and convergence. Given initial $(\mathbf{x}^{(0)}, \mathbf{y}^{(0)})$, the iterative algorithm updates $\mathbf{x}$ and $\mathbf{y}$ using:

$$\left[x_i^{(t+1)}\right]^\beta = x_i^{(t)} \frac{(B\mathbf{y}^{(t)})_i}{[\mathbf{x}^{(t)}]^T B \mathbf{y}^{(t)}}, \tag{25}$$

$$\left[y_j^{(t+1)}\right]^\beta = y_j^{(t)} \frac{(B^T\mathbf{x}^{(t)})_j}{[\mathbf{x}^{(t)}]^T B \mathbf{y}^{(t)}}, \tag{26}$$

When B is symmetric, this algorithm reduces to Eq. (19). The feasibility, optimality, and convergence of the algorithm can be established in similar manner as in the case of maximal clique.

Upper bound on the size of biclique.

We can establish a new theoretical bound on the size of biclique as a consequence of Theorem 8. In particular, we have:

$$|R_1 C_1|^{1/2} = \frac{(\mathbf{x}^*)^T B \mathbf{y}^*}{||\mathbf{x}^*|| \cdot ||\mathbf{y}^*||} \leq \max_{\mathbf{x},\mathbf{y}} \frac{\mathbf{x}^T B \mathbf{y}}{||\mathbf{x}|| \cdot ||\mathbf{y}||} = \sigma_1(B),$$

which yields the following upper bound:

$$|R_1 C_1| \leq \sigma_1^2(B). \tag{27}$$

4.3 Test on synthetic data

We test the ability of the algorithm to detect maximal bicliques. We embed a known biclique into the standard random graphs (two nodes are joined with an edge with fixed probability $p = 0.3$). We vary the size of the embedded cliques, while fixing the rectangular matrix to be 500×1000. We set $\alpha = \beta = 1.05$. We tested on 20 bipartite graphs with randomly generated adjacency matrices. The embedded maximal bicliques are readily detected in all cases.

4.4 Finding Bicliques in Document Collections

In this section, we apply our algorithm to discover bicliques in document collections. A document collection can be represented as a binary document-term matrix where each entry is 1 or 0 denoting whether the corresponding document and term co-occur or not. The document-term matrix can be expressed as a unweighted bipartite graph with a set of *document* nodes and a set of *term* nodes. If a document contains a term, an edge exists to connect them. Hence bicliques in binary document-term matrices are subsets of documents with tightly coupled terms. These bicliques represent specific topics and are explicitly supported/described by both representative subgroups of documents and representative subgroups of words.

4.4.1 Experiment Setup

Since each biclique is composed of representative documents and representative, purity measure can then be used to measure the extent to which each biclique contained documents from primarily one class [38].

$$Purity = \sum_{i=1}^{K} \frac{n_i}{n} P(S_i), P(S_i) = \frac{1}{n_i} max_j(n_i^j), \quad (28)$$

where S_i is a particular biclique or cluster with n_i documents, n_i^j is the number of documents of the j-th class that were assigned to the i-th biclique or cluster, K is the number of bicliques or clusters and n is the total number of extracted documents. In general, the larger the values of purity, the better the documents can describe the biclique.

We use five real world datasets described in Table 3 in our experiments. **CSTR** is a dataset of the abstracts of technical reports (TRs) published in the Department of Computer Science at a research university. **WebKB4** and **WebKB7** datasets are webpages gathered from university computer science departments. **Reuters** is a subset of Reuters-21578 Text Categorization Test collection that includes the 10 most frequent categories. **WebACE** is the dataset generated from WebACE project [14].

Datasets	# Documents	# words	# Classes
CSTR	476	1000	4
WebKB4	4199	1000	4
WebKB7	8280	1000	7
Reuter	2900	1000	10
WebAce	2340	1000	20

Table 3. Dataset Description.

It is important to note that cliques and bicliques discovered using our algorithm are not always 100% complete cliques — many computed cliques have missing edges. In general, the missing edges are less than 30% of the total possible edges in the computed psuedo cliques.

The clustering procedure is the following. We compute the bicliques one at a time. After a biclique is computed, the edges among the nodes are removed. The algorithm is then repeat once more to find the next biclique. This is repeated until the computed cliques is less than minimum size, which we set to 5.

4.4.2 Results

We compare our biclique finding algorithm with the traditional K-Means clustering, and several co-clustering algorithms including information theoretic co-clustering algorithm (ITCC) [7], Euclidean co-clustering algorithm (ECCC), and minimum squared residue co-clustering algorithm (MRCC) [4] on five real world datasets described in Table 3. The experimental results are shown in Figure 2. the computed bicliques are generally tight.

The experiments confirm this property of our algorithm. It showes that documents in each biclique have higher purity. In other words, our algorithm is able to extract more meaningful bicliques from document collections.

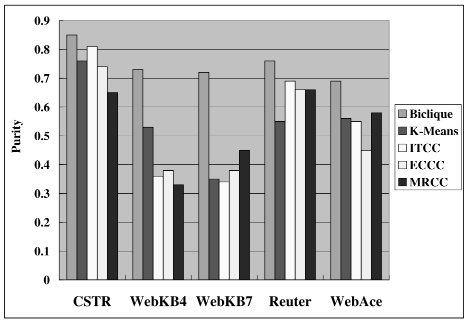


Figure 2. Purity Comparisons.

5 Conclusions

Recent progresses have shown nonnegative matrix factorization (NMF) provides a new versatile model for data clustering. In this paper, we propose several NMF inspired algorithms to solve different data mining problems including multi-way normalized cut spectral clustering, graph matching of both undirected and directed graphs, and maximum clique finding on both graphs and bipartite graphs. These NMF inspired algorithms are extremely simple to implement and their convergence can be rigorously proven. We conduct experiments to demonstrate the effectiveness of these new algorithms. This work highlights that techniques developed in machine learning could have much broader applicability.

Acknowledgments

C. Ding is partially supported by NSF grant DMS-0844497. T. Li is partially supported by NSF grants IIS-0546280 and DMS-0844513.

Appendix

Proof of Theorem 2.

We use the auxiliary function approach [18]. An auxiliary function $G(H, \tilde{H})$ of function $L(H)$ satisfies

$$G(H, H) = L(H), \quad G(H, \tilde{H}) \le L(H). \tag{29}$$

We define

$$H^{(t+1)} = \arg\max_{H} G(H, H^{(t)}). \tag{30}$$

Then by construction, we have

$$L(H^{(t)}) = G(H^{(t)}, H^{(t)}) \le G(H^{(t+1)}, H^{(t)}) \le L(H^{(t+1)}). \tag{31}$$

This proves that $L(H^{(t)})$ is monotonically increasing.

The key steps in the remainder of the proof are: (1) Find an appropriate auxiliary function; (2) Find the *global* maxima of the auxiliary function. It is important that the maxima in Eq. (30) are the global maxima, otherwise the first inequality of Eq. (31) does not hold.

We can show that

$$\begin{aligned} G(H, \tilde{H}) = \quad & \sum_{k} \sum_{ij} W_{ij} \tilde{H}_{ik} \tilde{H}_{jk} (1 + \log \frac{H_{ik} H_{jk}}{\tilde{H}_{ik} \tilde{H}_{jk}}) \\ - \quad & \sum_{i=1}^{p} \sum_{k,l} \frac{(D\tilde{H}\alpha)_{ik} H_{ik}^2}{\tilde{H}_{ik}} \end{aligned} \tag{32}$$

is an auxiliary function of $L(H)$ of Eq. (4) (the constant term $\mathrm{Tr}\,\alpha$ is ignored). Using the inequality $z \ge 1 + \log z$ and setting $z = H_{ik} H_{jk} / \tilde{H}_{ik} \tilde{H}_{jk}$, the first term in Eq. (32) is a lower bound of the first term in Eq. (4).

We note a generic inequality

$$\sum_{i=1}^{n} \sum_{p=1}^{k} \frac{(AS'B)_{ip} S_{ip}^2}{S'_{ip}} \ge \mathrm{Tr}(S^T ASB), \tag{33}$$

where $A > 0, B > 0, S > 0, S' > 0$, with A and B symmetric. Using this, we can see the second term in Eq. (32) is a lower bound of the second term in Eq. (4).

According to Eq. (30), we need to find the global maxima of $G(H, \tilde{H})$ for H. The gradient is

$$\frac{\partial G(H, \tilde{H})}{\partial H_{ik}} = 2 \frac{(W\tilde{H})_{ik} \tilde{H}_{ik}}{H_{ik}} - 2 \frac{(D\tilde{H}\alpha)_{ik} H_{ik}}{\tilde{H}_{ik}} \tag{34}$$

The second derivative is

$$\frac{\partial^2 G(H, \tilde{H})}{\partial H_{ik} \partial H_{j\ell}} = -2 [\frac{(W\tilde{H})_{ik} \tilde{H}_{ik}}{H_{ik}^2} + \frac{(D\tilde{H}\alpha)_{ik}}{\tilde{H}_{ik}}] \delta_{ij} \delta_{k\ell}. \tag{35}$$

Thus $G(H, \tilde{H})$ is a concave function in H and has a unique global maximum. This global maximum can be obtained by setting the first derivative to zero, which gives

$$H_{ik}^2 = \tilde{H}_{ik}^2 \frac{(W\tilde{H})_{ik}}{(D\tilde{H}\alpha)_{ik}}. \tag{36}$$

According to Eq. (30), $H^{(t+1)} = H$ and $H^{(t)} = \tilde{H}$. Thus we obtain the update rule in Eq. (3). □

References

[1] M. Berry, M. Browne, A. Langville, P. Pauca, and R. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *To Appear in Computational Statistics and Data Analysis*, 2006.

[2] I. Bomze, M. Budinich, P. Pardalos, and M. Pelillo. The maximum clique problem. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*, volume 4. Kluwer Academic Publishers, Boston, MA, 1999.

[3] J.-P. Brunet, P. Tamayo, T.R. Golub, and J.P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proc. Nat'l Academy of Sciences USA*, 102(12):4164–4169, 2004.

[4] H. Cho, I. Dhillon, Y. Guan, and S. Sra. Minimum sum squared residue co-clustering of gene expression data. In *Proceedings of The 4th SIAM Data Mining Conference*, pages 22–24, Lake Buena Vista, Florida, April 2004.

[5] M. Cooper and J. Foote. Summarizing video using non-negative similarity matrix factorization. In *Proc. IEEE Workshop on Multimedia Signal Processing*, pages 25–28, 2002.

[6] I. Dhillon and S. Sra. Generalized nonnegative matrix approximations with Bregman divergences. In *Advances in Neural Information Processing Systems 17*, Cambridge, MA, 2005. MIT Press.

[7] I. Dhillon, S. Mallela, and D. S. Modha. Information-theoretical co-clustering. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, pages 89–98, 2003.

[8] C. Ding and X. He. K-means clustering and principal component analysis. *Int'l Conf. Machine Learning (ICML)*, 2004.

[9] C. Ding, X. He, and H.D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. *Proc. SIAM Data Mining Conf*, 2005.

[10] C. Ding, T. Li, and W. Peng. Nonnegative matrix factorization and probabilistic latent semantic indexing: Equivalence, chi-square statistic, and a hybrid method. *Proc. National Conf. Artificial Intelligence*, 2006.

[11] E. Gaussier and C. Goutte. Relation between plsa and nmf and implications. In *Proc. of ACM SIGIR conference*, pages 601–602, New York, NY, USA, 2005. ACM Press.

[12] L. Gibbons, D. Hearn, P. Pardalos, and M. Ramana. Continuous characterizations of the maximum clique problem. *Mathematics of Operations Research*, 22:754–768, 1997.

[13] M. Gu, H. Zha, C. Ding, X. He, and H. Simon. Spectral relaxation models and structure analysis for k-way graph clustering and bi-clustering. *Penn State Univ Tech Report CSE-01-007*, 2001.

[14] E-H. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. WebACE: A web agent for document categorization and exploration. In *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*. ACM Press, 1998.

[15] P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *J. Machine Learning Research*, 5:1457–1469, 2004.

[16] Marek Karpinski and Wojciech Rytter. *Fast parallel algorithms for graph matching problems*. Oxford University Press, Inc., New York, NY, USA, 1998.

[17] D.D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[18] D.D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

[19] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

[20] S.Z. Li, X. Hou, H. Zhang, and Q. Cheng. Learning spatially localized, parts-based representation. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 207–212, 2001.

[21] T. Li. A general model for clustering binary data In *Proc. of KDD 2005*, pages 188–197, 2005.

[22] T. Li and S. Ma. IFD: Iterative feature and data clustering. In *Proc. SIAM Int'l conf. on Data Mining (SDM 2004)*, pages 472–476, 2004.

[23] T. Li, S. Ma and M. Ogihara. Document clustering via adaptive subspace iteration. In *Proc. of SIGIR*, pages 218–225, 2004.

[24] T.S. Motzkin and E.G. Straus. Maxima for graphs and a new proof of a theorem of turan. *Canad. J. Math.*, 17:533–540, 1965.

[25] A.Y. Ng, M.I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Proc. Neural Info. Processing Systems (NIPS 2001)*, 2001.

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

[27] V. P. Pauca, F. Shahnaz, M.W. Berry, and R.J. Plemmons. Text mining using non-negative matrix factorization. In *Proc. SIAM Int'l conf on Data Mining*, pages 452–456, 2004.

[28] M. Pelillo. Relaxation labeling networks for the maximum clique problem. *Journal of Artificial Neural Networks*, 2(4):313–328, 1995.

[29] F. Sha, L. K. Saul, and D. D. Lee. Multiplicative updates for nonnegative quadratic programming in support vector machines. In *Advances in Neural Information Processing Systems 15*. MIT Press, Cambridge, MA, 2003.

[30] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE. Trans. on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[31] N. Srebro, J. Rennie, and T. Jaakkola. Maximum margin matrix factorization. In *Advances in Neural Information Processing Systems*, Cambridge, MA, 2005. MIT Press.

[32] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 10:695 – 703, 1988.

[33] J. von Neumann. Some matrix inequalities and metrization of matric-spaces. *Tomsk Univ. Rev. Linear Algebra and Its Applications*, 1(7):286–300, 1937.

[34] Y.-L. Xie, P.K. Hopke, and P. Paatero. Positive matrix factorization applied to a curve resolution problem. *Journal of Chemometrics*, 12(6):357–364, 1999.

[35] W. Xu, X. Liu, and Y. Gong. Document clustering based on non-negative matrix factorization. In *Proc. ACM Conf. Research development in IR (SIGIR)*, pages 267–273, 2003.

[36] D. Zeimpekis and E. Gallopoulos. CLSI: A flexible approximation scheme from clustered term-document matrices. *Proc. SIAM Data Mining Conf*, pages 631–635, 2005.

[37] H. Zha, C. Ding, M. Gu, X. He, and H.D. Simon. Spectral relaxation for K-means clustering. *Advances in Neural Information Processing Systems 14 (NIPS'01)*, pages 1057–1064, 2002.

[38] Y. Zhao and G. Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55(3):311–331, 2004.