CASNMF: A Converged Algorithm for Symmetrical Nonnegative Matrix Factorization

Li-Ping Tian[☆a], Ping Luo[☆b], Haiying Wang[c], Huiru Zheng[c], Fang-Xiang Wu[d,e,*]

[a]*School of Information, Beijing Wuzi University, Beijing, 101149, China*
[b]*Division of Biomedical Engineering, University of Saskatchewan, Saskatoon, SK S7N5A9, Canada*
[c]*School of Computing and Mathematics, University of Ulster, Jordanstown campus, Newtownabbey, North Ireland, BT37 0QB, UK.*
[d]*School of Mathematical Sciences, Nankai University, Tianjin, 300071, China,*
[e]*Department of Mechanical Engineering and Division of Biomedical Engineering, University of Saskatchewan, Saskatoon, SK S7N5A9, Canada.*

Abstract

Nonnegative matrix factorization (NMF) is a very popular unsupervised or semi-supervised learning method useful in various applications including data clustering, image processing, and semantic analysis of documents. This study focuses on symmetric NMF (SYM-NMF), which is a special case of NMF and can be useful in network analysis. Although there exist several algorithms for SYM-NMF in literature, their convergence and initialization have not been well addressed. In this paper we first discuss the convergence and initialization of existing algorithms for SYM-NMF. We then propose a converged algorithm for SYM-NMF (called CASNMF) which minimizes the Euclidean distance between a symmetrical matrix and its approximation of SYM-NMF. Based on the optimization principle and the local auxiliary function method, we prove that our presented CASNMF does not only converge to a stationary point, but also could be applied to the wider range of SYM-NMF problems. In addition, CASNMF does not require that the initial values are nonzero. To verify our theoretical results, experiments on three data sets have been conducted by comparing our proposed CASNMF with other existing method.

☆Equally contributed authors
*Corresponding author
Email address: faw341@mail.usask.ca (Fang-Xiang Wu)

CASNMF: A Converged Algorithm for Symmetrical Nonnegative Matrix Factorization

Li-Ping Tian[☆a], Ping Luo[☆b], Haiying Wang[c], Huiru Zheng[c], Fang-Xiang Wu[d,e,*]

[a]*School of Information, Beijing Wuzi University, Beijing, 101149, China*
[b]*Division of Biomedical Engineering, University of Saskatchewan, Saskatoon, SK S7N5A9, Canada*
[c]*School of Computing and Mathematics, University of Ulster, Jordanstown campus, Newtownabbey, North Ireland, BT37 0QB, UK.*
[d]*School of Mathematical Sciences, Nankai University, Tianjin, 300071, China,*
[e]*Department of Mechanical Engineering and Division of Biomedical Engineering, University of Saskatchewan, Saskatoon, SK S7N5A9, Canada.*

Abstract

Nonnegative matrix factorization (NMF) is a very popular unsupervised or semi-supervised learning method useful in various applications including data clustering, image processing, and semantic analysis of documents. This study focuses on symmetric NMF (SYM-NMF), which is a special case of NMF and can be useful in network analysis. Although there exist several algorithms for SYM-NMF in literature, their convergence and initialization have not been well addressed. In this paper we first discuss the convergence and initialization of existing algorithms for SYM-NMF. We then propose a converged algorithm for SYM-NMF (called CASNMF) which minimizes the Euclidean distance between a symmetrical matrix and its approximation of SYM-NMF. Based on the optimization principle and the local auxiliary function method, we prove that our presented CASNMF does not only converge to a stationary point, but also could be applied to the wider range of SYM-NMF problems. In addition, CASNMF does not require that the initial values are nonzero. To verify our theoretical results, experiments on three data sets have been conducted by comparing our proposed CASNMF with other existing meth-

☆Equally contributed authors
*Corresponding author
Email address: faw341@mail.usask.ca (Fang-Xiang Wu)

ods.

Keywords: Symmetrical nonnegative matrix factorization (SYM-NMF), convergence, initialization, Karush–Kuhn–Tucher (KKT) optimality condition, stationary point, local auxiliary function

1. Introduction

In many situations, data to be analyzed can be arranged into a matrix, which is often called a data matrix. Therefore, some general matrix factorization methods, such as principle component analysis (PCA) or single value decomposition (SVD) [1, 2], independent component analysis (ICA) [3, 4], and network component analysis (NCA) [5, 6], have been widely used as very important unsupervised learning tools in discovering the underlying structures of data. Furthermore, many data such as image pixel data [7], text association data [8], and protein-protein interaction data [9] can be arranged into a nonnegative matrix whose elements are nonnegative. The general factorizations of nonnegative matrices by SVD, ICA and NCA contain negative entries and thus have difficulties for interpretation. Therefore, nonnegative matrix factorization (NMF) [10] has advantages over general factorizations. In contrast to those general factorizations, the nonnegativity in NMF ensures factors contain coherent parts of original data [11, 12].

In recent years, NMF has become a very popular unsupervised or semi-supervised learning method useful in various applications including data clustering [13, 14, 15], image processing [16, 17, 18], semantic analysis of documents [19, 20, 21], and biological data analysis [22, 23, 24, 25]. One of the main reasons that NMF has become popular is that Lee and Seung [10, 26] proposed a very simple and computationally efficient multiplicative update algorithm for solving NMF. However, Lee and Seung's algorithm has several weaknesses. To address these weaknesses, Lin proposed a few improved methods [27, 28], which yet converge slower than Lee and Seung's algorithm. Recently, Li et al. [29] proposed a fast multiplicative update algorithm for solving the NMF, which overcomes the drawbacks of both Lee and Seung's algorithm and Lin's algorithm by guaranteeing the convergence to a stationary point with some zero values for the initialization. Besides algorithms for basic NMF, algorithms are also proposed for constrained NMF, structured NMF, and generalized NMF. A comprehensive reviews about these algorithms can be found in [30].

This study focuses on symmetric NMF (SYM-NMF), which is a special case of constrained NMF. SYM-NMF tries to find a nonnegative matrix with a reduced rank to approximate a given symmetrical nonnegative data matrix. Mathematically, given a symmetrical nonnegative matrix A with the size of $n \times n$, and a positive integer $r < n$, SYM-NMF finds a nonnegative matrix $U = (u_{ik}) \in R^{n \times r}$ such that

$$A \approx UU^T.$$

Although there are several ways to measure the difference between A and its approximation UU^T such as the generalized Kullback-Leibler divergence between A and UU^T [10], this study adopts the Euclidean distance between A and UU^T in terms of the matrix Frobenius norm as follows

$$\begin{aligned} \min_U f(U) = \min_U \frac{1}{2}\left\|A - UU^T\right\|_F^2, \\ subject\ to\ u_{ik} \geq 0,\ \forall i, k, \end{aligned} \tag{1}$$

where $\left\|X\right\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}^2}$ is the Frobenius norm of a matrix $X = (x_{ij}) \in R^{n \times m}$. Note that each nonnegative constraint is subject to only a single variable. Furthermore, the objective function in (1) can also be expressed in terms of matrix elements as follows

$$f(U) = \frac{1}{2}\left\|A - UU^T\right\|_F^2 = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij} - \sum_{s=1}^{r} u_{is}u_{js})^2. \tag{2}$$

Although SYM-NMF is a special case of NMF, the algorithms for general NMF cannot be directly applied to solve SYM-NMF [31]. Actually several algorithms for SYM-NMF have been proposed [31, 32, 33, 34]. However, their convergence and/or initialization have not been well addressed. In addition, these algorithms assume that matrix A in (1) is positive definite in order that the convergence can be guaranteed. However, in practice, many nonnegative symmetric matrices are not positive definite, for example, the adjacent matrices of networks [35, 36].

In this study we propose a converged algorithm for SYM-NMF (CASNMF) and prove that our CASNMF does not only converge to a stationary point, but also could apply to the wider range of SYM-NMF problems as it does not require that A is positive definite for convergence. Moreover, CASNMF does not require that the initial values of matrix U are nonzero, which

is very useful in applications of SYM-NMF to semi-supervised machine learning [13, 14]. In Section 2, we first describe the Karush–Kuhn–Tucher (KKT) optimality conditions of optimization problem (1), with which a stationary point should be satisfied. Then, we review several existing algorithms for solving optimization problem (1) and discuss their relationships and convergence. In Section 3, we describe our new algorithm CASNMF for solving optimization problem (1), analyze its properties, and prove its convergence by using local auxiliary functions [37]. We also compare our CASNMF with the best algorithms reviewed in Section 2. To verify our theoretical results, in Section 4 we apply our proposed CASNMF algorithm, as well as the best existing algorithms reviewed in Section 2 to three data sets. The experimental results are analyzed and discussed. In Section 5, we conclude this study and discuss some directions of future work. The novelties of study include: 1) CASNMF is applicable to non-positive definite matrices (especially network adjacent matrices); 2) CASNMF can guarantee to converge to a local minimum stationary point; 3) initial values are not necessarily non-zeros; 4) CASNMF converges to a smaller local minimum, compared to the best existing algorithms.

2. KKT optimality conditions and existing algorithms for SYM-NMF

Although the optimization problem (1) is not convex in U, a local optimal solution of U should be a stationary point according to the optimization principle [38]. By the definition U is a stationary point of the optimization problem (1) if it satisfies the following KKT optimality conditions [38]:
For $\forall i, k$

$$u_{ik} \geq 0, \tag{3a}$$

$$\frac{\partial f(U)}{\partial u_{ik}} \geq 0, \tag{3b}$$

$$and \quad u_{ik}\frac{\partial f(U)}{\partial u_{ik}} = 0, \tag{3c}$$

where

$$\frac{\partial f(U)}{\partial u_{ik}} = 2\sum_{m=1}^{n}(\sum_{s=1}^{r}(u_{ms}u_{is}) - a_{mi})u_{mk}, \tag{4}$$

is the partial derivatives of $f(U)$ with respect to u_{ik}, seeing Appendix A for its derivation.

To solve the optimization problem (1), Zass and Shashua [31] proposed the following Algorithm 1.

Algorithm 1. *Zass and Shashua's Algorithm [31]*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^{t} \frac{(AU^t)_{ik} - a_{ii}u_{ik}^t}{[U^t(U^t)^T U^t]_{ik} - [U^t(U^t)^T]_{ii}u_{ik}^t}, \quad \forall i,k. \tag{5}$$

By ignoring the diagonal elements of matrix A, they proved [31] that the objective function of problem (1) was non-increasing with the update rule (5). When converging, $U^t(U^t)^T$ tends to A and the denominator tends to the numerator in formula (5). However, they did not prove that the solution converges to a stationary point [31]. In addition, it is unreasonable to simply ignore the diagonal elements of matrix A in (1) for making the problem manageable.

Long et al. empirically proposed the following Algorithm 2 [32] without a rigorous derivation. Actually Long's algorithm can be viewed as a revision of Lee and Seung's algorithm [10, 29] for the general NMF by forcing that the second matrix is the transpose of the first matrix.

Algorithm 2. *Long's 2005 Algorithm [32]*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^{t} \frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}}, \quad \forall i,k. \tag{6}$$

The update rule (5) can also be viewed as the revised version of update rule (6) by neglecting the diagonal elements of matrices A and UU^T in problem (1). Later on, Long et al. [33] proposed another algorithm (Algorithm 3) with a rigorous derivation.

Algorithm 3. *Long's 2007 Algorithm [33]*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^{t} \sqrt[4]{\frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}}}, \quad \forall i,k. \tag{7}$$

Long et al. [33] proved that Algorithm 3 guaranteed that the objective function of problem (1) was non-increasing with the increase of iterations. However, they did not prove that Algorithm 3 converges to a stationary point. He et al. [34] proposed an algorithm (Algorithm 4) to improve Algorithm 3 and proved that their algorithm would converge to a stationary point if matrix A is definite positive.

Algorithm 4. *He's Algorithm [34]*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^{t} \sqrt[3]{\frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}}}, \quad \forall i,k. \tag{8}$$

He et al. [34] formulated the update rules (7) and (8) as the *α-weighted geometric mean* between u_{ik} and updated rule (6) by the following formula

$$\begin{aligned} u_{ik}^{t+1} &= (u_{ik}^{t})^{1-\alpha} \left[u_{ik}^{t} \frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}} \right]^{\alpha} \\ &= u_{ik}^{t} \left[\frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}} \right]^{\alpha}, \end{aligned} \tag{9}$$

where $0 \leq \alpha \leq 1$. The rules (6)–(8) are the formula (9) with $\alpha = 1$, $1/4$ *and* $1/3$, respectively. Furthermore, He et al. [34] proved that the larger the value of α, the faster the solution of problem (1) with the rule (9) converges. However they illustrated by simulation that Algorithm 2 would not converge to a local minimum as the objective function of problem (1) is oscillating with the increase of iterations. Therefore, it is suggested that $\alpha = 0.99$ should be used.

Ding et al. [12] proposed the following algorithm without a rigorous derivation.

Algorithm 5. *Ding's Algorithm [12]*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^{t} \left(1 - \beta + \beta \frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}} \right), \quad \forall i,k, \tag{10}$$

where $0 < \beta \leq 1$. Ding et al. suggested that in practice $\beta = 1/2$ should be a good choice [12, 21]. However, the convergence of this algorithm has

not been proved yet. It is obvious that the update rule (10) becomes the update rule (6) when $\beta = 1$. He et al. [34] formulated the update rule (10) as the *β-weighted arithmetic mean* between u_{ik} and the update rule (6) by the following formula

$$\begin{aligned} u_{ik}^{t+1} &= (1-\beta)u_{ik}^t + \beta u_{ik}^t \frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}} \\ &= u_{ik}^t \left(1 - \beta + \beta \frac{(AU^t)_{ik}}{[U^t(U^t)^T U^t]_{ik}}\right). \end{aligned} \tag{11}$$

Furthermore, He et al. [34] proved that the larger the value of β, the faster the solution of problem (1) with the rule (11) converges. As Algorithm 2 does not converge to a local minimum as the objective function of problem (1) is oscillating with the increase of iterations [34], it is suggested that $\beta = 0.99$ should be used.

Actually descent methods are widely used to find a local minimum of an objective function [38]. Typically descent methods need to define a search step size and a search direction such that the value of the objective function is decreasing (at least non-increasing) with increasing the number of iterations. As it is the steepest descent direction for minimizing the objective function, the negative gradient is naturally chosen as the search direction. The resulting methods are called the gradient descent algorithms. Applying the gradient descent method to the optimization problem (1) yields to the gradient descent algorithm (Algorithm 6), where $\varepsilon(u_{ik}^t)$ is the search step size at iteration t.

Algorithm 6. *Gradient descent algorithm*
For t = 1,2,...

$$u_{ik}^{t+1} = u_{ik}^t - \varepsilon(u_{ik}^t) \frac{\partial f(U^t)}{\partial u_{ik}^t}, \quad \forall i, k. \tag{12}$$

In principle, on the one hand, the small positive search step sizes $\varepsilon(u_{ik}^t)$ can warrant that the value of objective function is decreasing (at least non-increasing) with increasing the number of iterations. However, it would be very slow to converge with the small search step sizes. On the other hand, the larger the search step size, the faster the value of objective function is decreasing, but the updated variables may not satisfy the constraints. Therefore, although the negative gradient descent is known as the steepest

descent, the gradient descent algorithm could very slowly converge to a local minimum or even not converge to any local minimum without a smart choice of search step sizes, especially for the optimization problems with constraints such as optimization problem (1).

By taking the following search step size

$$\varepsilon_{Ding}(u_{ik}^t) = \frac{\beta u_{ik}^t}{2[U^t(U^t)^T U^t]_{ik}}, \quad \forall i, k. \tag{13}$$

we can obtain the update rule (10) of Algorithm 5 for $0 < \beta \leq 1$ and thus the update rule (6) of Algorithm 2 with $\beta = 1$ in (13). As a result, Algorithms 1, 2 and 5 can be viewed as the special case of the gradient descent algorithm (Algorithm 6) while Algorithms 3 and 4 can be viewed as some types of "mean" between u_{ik} and the gradient descent algorithm. From (13), we can see that the larger the value of β, the larger the search step size is and thus the faster the solution of problem (1) with the rule (11) converges, which is in agreement with the conclusion in [34].

It can be seen that all Algorithms 1–5 can guarantee that all updated elements of matrix U are nonnegative if all initial values are nonnegative. However, one can observe that the update rules in these algorithm contain a factor u_{ik}. Therefore, if at some iteration step t_0, it turns out $u_{ik}^t = 0$, the value of u_{ik} will have no chance to be updated for all iterations $t > t_0$ even though the KKT conditions (3b) and (3c) are not satisfied yet. As a result, these algorithms cannot converge to a stationary point even if they converge. With this observation, the elements of matrix U for Algorithms 1–5 should not be initialized as 0. However, in some applications of SYM-NMF such as semi-supervised machine learning for clustering [13, 14], one may have to initialize some elements of U as 0. In addition, these algorithms are not well defined if denominators in the update rules (5-10) are zero. Although a small positive number can be added to the denominator as in [27], the convergence speed may slow down.

Furthermore, among all these algorithms, Algorithm 2 may not converge as simulated in [34]. The convergence of Algorithm 5 was not provided. Although Algorithms 1 and 3 can be proved to converge in [31, 33], they were not provided with the proof to converge to a stationary point. Algorithm 4 is the only one that was proved to converge to a stationary point. However, to guarantee its convergence, Algorithm 4 assumes that matrix A is positive definite and the initial values are nonzero [34]. Actually, these assumption are

too strong in many applications of SYM-NMF. For example, applying SYM-NMF for clustering nodes based on the adjacent matrix of a network [35].

3. CASNMF and its convergence

Before describing our new algorithm CASNMF, we would like to discuss about the parameter r in SYM-NMF, which is the number of columns of matrix U. In most applications, one may know the range of values of r if not knowing its exact value. In some situations, one may want to find the minimal value of r such that the approximation of SYM-NMF is good enough. Under no circumstances, one expects the resultant matrix U has a zero column. In practice, if these situations happen, one can reduce the value of r by removing zero columns in U.

3.1. New update algorithm for SYM-NMF

To address the weaknesses of Algorithms 1–5, we propose the following update algorithm for solving the optimization problem (1).

Algorithm 7. *CASNMF for SYM-NMF*

1) Initialize $u_{ik}^1 \geq 0$, for all i, k,
2) For t = 1,2,...
   If U^t is stationary, i.e., satisfies KKT optimality conditions (3), then stop
   Else
   If $\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t = 0$,

$$u_{ik}^{t+1} = \sqrt{b_i^t}, \quad \forall i, k, \tag{14}$$

   Else

$$u_{ik}^{t+1} = max(0, \overline{u}_{ik}^{t+1}), \quad \forall i, k, \tag{15}$$

where

$$b_i^t = a_{ii} - \sum_{s=1}^{r} (u_{is}^t)^2, \tag{16}$$

$$\overline{u}_{ik}^{t+1} = u_{ik}^t - \frac{1}{2\left(\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t\right)} \frac{\partial f(U^t)}{\partial u_{ik}^t}, \tag{17}$$

$$D_{ik}^t = max[0, -b_t^t + (u_{ik}^t)^2 + 2u_{ik}^t d^t + (d^t)^2/2], \tag{18}$$

$$d^t = \frac{|\sum_{m=1}^{n}(\sum_{s=1}^{r}(u_{ms}^t u_{is}^t) - a_{mi})u_{mk}^t|}{\sum_{m=1}^{n}(u_{mk}^t)^2}. \tag{19}$$

As can be seen, different from Algorithms 1–5, the search step sizes of our newly proposed Algorithm 7 are

$$\varepsilon_{Wu}(u_{ik}^t) = \frac{1/2}{\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t}. \tag{20}$$

Ding's algorithm with $\beta = 1$ in (13) is the same as Long's 2005 algorithm and both of them converge the fastest among Algorithms 1–5. In this case, we have the following search step size.

$$\begin{aligned} \varepsilon_{Ding}(u_{ik}^t) &= \frac{u_{ik}^t}{2[U^t(U^t)^T U^t]_{ik}} = \frac{u_{ik}^t}{2\sum_{m=1}^{n}\sum_{s=1}^{r} u_{is}^t u_{ms}^t u_{mk}^t} \\ &= \frac{(1/2)u_{ik}^t}{\sum_{m=1}^{n}\sum_{\substack{s=1 \\ s\neq k}}^{r} u_{is}^t u_{ms}^t u_{mk}^t + \sum_{m=1}^{n}(u_{mk}^t)^2 u_{ik}^t} = \frac{1/2}{\sum_{m=1}^{n}(u_{mk}^t)^2 + E_{ik}^t}, \end{aligned} \tag{21}$$

where $E_{ik}^t = \sum_{m=1}^{n}\sum_{\substack{s=1 \\ s\neq k}}^{r} u_{is}^t u_{ms}^t u_{mk}^t / u_{ik}^t$. In principle, if E_{ik}^t is larger than D_{ik}^t, we can conclude that CASNMF (Algorithm 7) converges faster than Algorithms 1–5. Although it is not easy to say whether E_{ik}^t is larger than D_{ik}^t for all iterations t, it is true that E_{ik}^t is larger than D_{ik}^t for many cases when t is large. Actually, as $t \to \infty$ (i.e., after enough iterations), we have that $\sum_{s=1}^{r}(u_{is}^t)^2 - a_{ii} \to 0$ and $d^t \to 0$. As a result, we have that $D_{ik}^t \to (u_{ik}^t)^2$ as $t \to \infty$. Therefore, when u_{ik}^t is smaller, our algorithm will converge faster. On the other hand, as in most cases $\sum_{m=1}^{n}\sum_{\substack{s=1 \\ s\neq k}}^{r} u_{is}^t u_{ms}^t u_{mk}^t$ is strict positive, E_{ik}^t will be larger when u_{ik}^t is smaller. As a result, Algorithms 2 and 5 will converge slower. In applications, the optimal matrix U typically has many zeros or small elements [10, 12, 29, 30]. Therefore, our algorithms should converge faster than Algorithms 1 to 5 in practices.

Note that when $u_{ik}^t = 0$, and $\partial f(U^t)/\partial u_{ik}^t < 0$, the (i,k)-th element of U still has a chance to be updated and the updated value of u_{ik}^{t+1} is positive in CASNMF. However in Algorithms 1–5, as long as $u_{ik}^t = 0$ for some t, then for $t+1, t+2, \ldots$, the (i,k)-th element of U has no chance anymore to be updated even if $\partial f(U^t)/\partial u_{ik}^t < 0$ is true, which violates the KKT optimality condition (3b). Therefore, different from initializations of Algorithms 1–5, the initial values of elements of matrix U could be zeros for CASNMF. However, if all elements in a whole column of U are zero at iteration t, we may have $\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t = 0$. Therefore, u_{ik} will be updated by (14). Furthermore, if $a_{ii} = 0$ for all i, the elements in the k-th column of U will actually have no chance to be updated by (14) and (16). To avoid this case happened at the beginning, the initial matrix U should have no zero column. If matrix U would have a zero column when the algorithm converges, the zero column should be removed although all elements of zero column also satisfy all KKT optimality conditions (3). Note that the minimum value of objective function (1) will not be affected after the zero column is removed.

For all Algorithms 1–5, the main cost is dominated by computing matrix multiplications (AU^t) and $[U^t(U^t)^TU^t]$, each of which takes the order of $O(n^2r)$ operations at each iteration for updating all elements of U. Therefore, the computational complexity order of all Algorithms 1–5 is #iterations $*O(n^2r)$. For our presented Algorithm 7, the main cost for updating one element u_{ik}^t is dominated by computing $\partial f(U^t)/\partial u_{ik}^t = \sum_{m=1}^{n}(\sum_{s=1}^{r}(u_{ms}^t u_{is}^t) - a_{mi})u_{mk}^t$, which takes the order of $O(nr)$ operations. As a result, at each iteration for updating all elements of U, it takes the order of $O(n^2r^2)$ operations. Therefore, the computational complexity order of presented Algorithm 7 is #iterations $*O(n^2r^2)$. As typically $r \ll n$ is true and our algorithm needs much less number of iterations, the computational complexity of our presented Algorithm 7 would not be increased greatly, compared to Algorithms 1–5.

3.2. Proof of convergence

An auxiliary function is required for the proof of convergence of CASNMF like in the case of general NMF [26, 27, 28, 29]. For the general NMF, the objective function is quadratic in the elements of two factor matrices and thus the second derivatives of the objective function with respect to the elements of two factor matrices are constant and the higher derivatives are zeros [29]. However, for the SYM-NMF, the objective function is a fourth degree polynomial in the elements of the factor matrix U and thus the second derivatives

of the objective function with respect to the elements of the factor matrix are still a function of the elements of the factor matrix U (see Appendix A). Although a global auxiliary function was proposed in [34] for the SYM-NMF, it requires that matrix A is positive definite. In this study, different from the auxiliary functions used in [26, 27, 28, 29] for the convergences of algorithms for the general NMF or the one used in [34] for the convergence of algorithms for the special SYM-NMF, we introduce the concept of local auxiliary function and its important property below.

Definition: Let F be a function of a single variable u and G be a function of two variables u and $u^{'}$, where u is any real number while $|u-u^{'}| <= d$ and d is a nonnegative number. $G(u,u')$ is a local auxiliary function for $F(u)$ if the following conditions are true

$$G(u,u) = F(u) \text{ and } G(u,u') \geq F(u). \tag{22}$$

The concept of local auxiliary function is very useful in proof of convergence because of the following property.

Lemma 1. *If $G(u,u')$ is a local auxiliary function for $F(u)$, then F is non-increasing under the following update*

$$u^{t+1} = \arg \min_{u,|u-u^t|\leq d^t} G(u,u^t), \tag{23}$$

and $|u^{t+1}-u^t| \leq d^t$ where $d^t(\leq d)$ is a nonnegative number.

The proof is straightforward by considering

$$F(u^{t+1}) \leq G(u^{t+1},u^t) \leq G(u^t,u^t) = F(u^t). \tag{24}$$

To prove the convergence of their algorithm (Algorithm 4) for SYM-NMF [34], He et al. designed a global auxiliary functions for the objective function (1). However, two assumptions are required to prove the convergence of their algorithm: (1) matrix A should be definite positive and (2) every element of matrix U^t at any t should be nonzero. These two strong assumptions make their algorithm inapplicable to many practical problems.

Theorem 2. *With the update rules of our proposed CASNMF, the objective function of problem (1) is non-increasing with the increase of iterations and actually is decreasing as fast as possible because the largest possible search step was chosen at each iteration.*

PROOF. In the following, we derive the formulas for updating the (i,k)-th element of U in our presented Algorithm 7. Given a matrix U^t, we would like to understand how the objective function of optimization problem (1) changes when only the (i,k)-th element u_{ik} of U^t is deviated from U^t. By Taylor series, the objective function of optimization problem (1) can be expended in the (i,k)-th element of U as follows

$$\begin{aligned}
&f(U)\\
&= f(U^t) + \frac{\partial f(U^t)}{\partial u_{ik}^t}(u_{ik} - u_{ik}^t) + \frac{1}{2}\frac{\partial^2 f(U^t)}{\partial (u_{ik}^t)^2}(u_{ik} - u_{ik}^t)^2\\
&+ \frac{1}{6}\frac{\partial^3 f(U^t)}{\partial (u_{ik}^t)^3}(u_{ik} - u_{ik}^t)^3 + \frac{1}{24}\frac{\partial^4 f(U^t)}{\partial (u_{ik}^t)^4}(u_{ik} - u_{ik}^t)^4\\
&= f(U^t) + \frac{\partial f(U^t)}{\partial u_{ik}^t}(u_{ik} - u_{ik}^t) + \frac{1}{2}\frac{\partial^2 f(U^t)}{\partial (u_{ik}^t)^2}(u_{ik} - u_{ik}^t)^2\\
&+ 2u_{ik}^t(u_{ik} - u_{ik}^t)^3 + \frac{1}{2}(u_{ik} - u_{ik}^t)^4\\
&= f(U^t) + \frac{\partial f(U^t)}{\partial u_{ik}^t}(u_{ik} - u_{ik}^t) +\\
&\frac{1}{2}\Bigg[2\Bigg(\sum_{m=1}^{n} u_{mk}^2 + \sum_{s=1}^{r} u_{is}^2 + u_{ik}^2 - a_{ii}\Bigg) +\\
&4u_{ik}^t(u_{ik} - u_{ik}^t) + (u_{ik} - u_{ik}^t)^2\Bigg](u_{ik} - u_{ik}^t)^2.
\end{aligned} \tag{25}$$

Here, we define the following function with respect to u_{ik}

$$\begin{aligned}
G(U, U^t) = f(U^t) &+ \frac{\partial f(U^t)}{\partial u_{ik}^t}(u_{ik} - u_{ik}^t)\\
&+ \Bigg[\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\Bigg](u_{ik} - u_{ik}^t)^2,
\end{aligned} \tag{26}$$

where

$$D_{ik}^t = \max[0, -b_i^t + (u_{ik}^t)^2 + 2u_{ik}^t d^t + (d^t)^2/2]. \tag{27}$$

From (26), we can see that $G(U, U^t) = f(U)$ when $U^t = U$. In addition, for any $\alpha_{ik} \geq 0$, the third term in (26) is greater than or equal to the third term in (25) if $|u_{ik} - u_{ik}^t| \leq d^t$ while the first two terms in (25) and (26) are the same. Therefore, by the definition of the local auxiliary function, for any

$\alpha_{ik} \geq 0$, function defined in (26) is the local auxiliary function for function (25) if $|u_{ik} - u_{ik}^t| \leq d^t$. Minimizing function (26) with respect to u_{ik} yields the following update rule for u_{ik}:

$$\overline{\overline{u}}_{ik}^{t+1} = u_{ik}^t - \frac{1}{2\left(\sum_{m=1}^n (u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\right)} \frac{\partial f(U^t)}{\partial u_{ik}^t}, \tag{28}$$

for any positive value of d^t. Furthermore, if $|\overline{\overline{u}}_{ik}^{t+1} - u_{ik}^t| \leq d^t$ for some d^t , then we have

$$\begin{aligned} &\arg \min_{u_{ik}, |u_{ik} - u_{ik}^t| \leq d^t} G(U, U^t) \\ &= u_{ik}^t - \frac{1}{2\left(\sum_{m=1}^n (u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\right)} \frac{\partial f(U^t)}{\partial u_{ik}^t}. \end{aligned} \tag{29}$$

Actually from (28), we have

$$\begin{aligned} |\overline{\overline{u}}_{ik}^{t+1} - u_{ik}^t| &= \frac{1}{2\left(\sum_{m=1}^n (u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\right)} \left| \frac{\partial f(U^t)}{\partial u_{ik}^t} \right| \\ &\leq \frac{\left| \sum_{m=1}^n (\sum_{s=1}^r (u_{ms}^t u_{is}^t) - a_{mi}) u_{mk}^t \right|}{2 \sum_{m=1}^n (u_{mk}^t)^2}. \end{aligned} \tag{30}$$

Therefore, if the value of d^t is computed by (19), we can obtain the formula (28).

By Lemma 1, replacing u_{ik}^t with $\overline{\overline{u}}_{ik}^{t+1}$ calculated by (19), (27) and (28) for any $\alpha_{ik} \geq 0$ and keeping other elements in U^t unchanged, the value of objective function in the optimization problem (1) is non-increasing.

Note that when $\alpha_{ik} = 0$, formula (28) becomes formula (17), that is $\overline{\overline{u}}_{ik}^{t+1} = \overline{u}_{ik}^{t+1}$. The update rule (28) is actually a gradient descent rule with the search step size.

$$\varepsilon(u_{ik}^t) = \frac{1}{2\left(\sum_{m=1}^n (u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\right)}. \tag{31}$$

The larger the value of $\varepsilon(u_{ik}^t)$, the faster the value of the objective function in (1) is decreasing. For the given matrix U^t, the largest value of $\varepsilon(u_{ik}^t)$ is

$1/2\big(\sum_{m=1}^{n}(u_{mk}^t)^2 + 2D_{ik}^t\big)$ at $\alpha_{ik} = 0$. However, when the value of $\varepsilon(u_{ik}^t)$ is too large, the updated value of $\overline{\overline{u}}_{ik}^{t+1}$ by (28) could be negative, which violates the KTT optimality condition (3a). Therefore, we need to carefully choose the value of α_{ik} to make sure that all KKT optimal conditions (3) would be satisfied while the value of the objective function in (1) is decreasing as fast as possible, which means that we should choose the largest possible search step size at each iteration.

Here we consider two cases: (I) $\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t = 0$ and (II) $\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t \neq 0$. For case (I), we have $u_{ik}^t = 0$ for all $i = 1, \ldots, n$ and thus $\sum_{s=1}^{r}(u_{is}^t)^2 - a_{ii} = -b_i^t \leq 0$. Furthermore, from equations (A2)–(A4) in Appendix A we have

$$\frac{\partial f(U^t)}{\partial u_{ik}^t} = 0, \quad \frac{\partial^2 f(U^t)}{\partial (u_{ik}^t)^2} = 2\left(\sum_{s=1}^{r}(u_{is}^t)^2 - a_{ii}\right) = -2b_i^t,$$
$$\frac{\partial^3 f(U^t)}{\partial (u_{ik}^t)^3} = 0, \quad \frac{\partial^4 f(U^t)}{\partial (u_{ik}^t)^4} = 12.$$

Thus the function $f(U)$ in (25) becomes

$$\begin{aligned} f(U) &= f(U^t) - b_i^t u_{ik}^2 + u_{ik}^4/2 \\ &= f(U^t) - (b_i^t)^2/2 + (u_{ik}^2 - b_i^t)^2/2. \end{aligned} \tag{32}$$

As $\partial f(U^t)/\partial u_{ik}^t = 0$, the KKT optimality conditions are satisfied for any nonnegative u_{ik}. Furthermore, from (32) when $u_{ik} = \sqrt{b_t^t}$, the function $f(U)$ reaches the minimum. Therefore by using formula (14) to calculate u_{ik}^{t+1}, the values of objective function is decreasing as fast as possible.

For case (II), we need to carefully choose the nonnegative α_{ik} such that the updated value of $\overline{\overline{u}}_{ik}^{t+1}$ calculated by (28) is nonnegative and the value of the objective function in (1) is decreasing as fast as possible. We need to consider two cases again: (a) $\partial f(U^t)/\partial u_{ik}^t < 0$ and (b) $\partial f(U^t)/\partial u_{ik}^t \geq 0$. For case (a), the updated value of $\overline{\overline{u}}_{ik}^{t+1}$ calculated by (28) is positive for any value of nonnegative value of α_{ik}. Thus we take $\alpha_{ik} = 0$ to have the largest search step size $1/\big(2\sum_{m=1}^{n}(u_{mk}^t)^2 + 2D_{ik}^t\big)$ and then formula (28) becomes (17). For case (b), the inequality

$$\frac{1}{2\left(\sum_{m=1}^{n}(u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\right)} \frac{\partial f(U^t)}{\partial u_{ik}^t} \geq 0, \tag{33}$$

is true. If $u_{ik}^t = 0$, we can choose $\alpha_{ik} = +\infty$ in (28) to get $u_{ik}^{t+1} = 0$. We can also get $u_{ik}^{t+1} = 0$ by formula

$$u_{ik}^{t+1} = \max(0, \overline{u}_{ik}^{t+1}),$$

as $\overline{u}_{ik}^{t+1}$ calculated by (17) is non-positive. If $u_{ik}^t > 0$, we should choose α_{ik} to make sure that the inequality

$$u_{ik}^t - \frac{1}{2\Big(\sum_{m=1}^n (u_{mk}^t)^2 + D_{ik}^t + \alpha_{ik}\Big)} \frac{\partial f(U^t)}{\partial u_{ik}^t} \geq 0, \tag{34}$$

is true. Solving (34) and considering $\alpha_{ik} \geq 0$ yields

$$\alpha_{ik} \geq \max(0, \frac{1}{2u_{ik}^t} \frac{\partial f(U^t)}{\partial u_{ik}^t} - \sum_{m=1}^n (u_{mk}^t)^2 - D_{ik}^t).$$

To have the largest possible search step size, we take the minimum value of α_{ik} as follows

$$\alpha_{ik} = \max(0, \frac{1}{2u_{ik}^t} \frac{\partial f(U^t)}{\partial u_{ik}^t} - \sum_{m=1}^n (u_{mk}^t)^2 - D_{ik}^t).$$

If $\alpha_{ik} = 0$, the value of $\overline{\overline{u}}_{ik}^{t+1}$ calculated by (27) (i.e. (17)) is nonnegative. If $\alpha_{ik} = \frac{1}{2u_{ik}^t} \frac{\partial f(U^t)}{\partial u_{ik}^t} - \sum_{m=1}^n (u_{mk}^t)^2 - D_{ik}^t > 0$, then the value of $\overline{\overline{u}}_{ik}^{t+1}$ calculated by (28) is exactly 0 while the value of $\overline{u}_{ik}^{t+1}$ calculated by (17) is nonpositive. Therefore, again we can have

$$u_{ik}^{t+1} = \max(0, \overline{u}_{ik}^{t+1}).$$

In summary, for case (b), we have the formulas (15) and (17)–(19) to update the value of u_{ik} which can guarantee that the updated value of u_{ik}^{t+1} is nonnegative and the value of the objective function (1) is non-increasing with the increase of iterations and actually is decreasing as fast as possible because the largest possible search step was chosen at every iteration.

In the following we prove that the sequence of matrices $\{U^t, t = 1, 2, \dots\}$ produced by CASNMF has at least one limit point which is also a stationary point.

Theorem 3. *The sequence of matrices $\{U^t, t = 1, 2, \dots\}$ produced by CASNMF has at least one limit point.*

PROOF. By Theorem 2, the sequence $\{f(U^t), t = 1, 2, \dots\}$ is non-increasing. As function $f(U)$ is bounded below, $\lim_{t\to\infty} f(U^t)$ exists and is finite. As matrix A is bounded, $\left\|U^t(U^t)^T\right\|_F^2$ is bounded for all $t \geq 1$. Furthermore, from Inequality (B.1) in Appendix B, the sequence $\{U^t, t = 1, 2, \dots\}$ is bounded in matrix Frobenius norm for all $t \geq 1$. Thus $\{U^t, t = 1, 2, \dots\}$ has at least one limit point.

Theorem 4. *Assume that $\{U^t, t = 1, 2, \dots\}$ is produced by CASNMF and converges to U^*. Then U^* satisfies KKT optimality conditions (3).*

PROOF. If U^* satisfies $\sum_{m=1}^{n}(u_{mk}^*)^2 + D_{ik}^* = 0$ for some i and k, we have $u_{mk}^* = 0$ for $m = 1, \dots, n$, which means the k-th column of U^* are zero. Although all the elements in the k-th column of U^* satisfy KKT conditions, it should be removed from the optimal solution as discussed before.

On the other hand, if U^* satisfies $\sum_{m=1}^{n}(u_{mk}^*)^2 + D_{ik}^* \neq 0$ for some i and k, from Algorithm 7, we have

$$u_{ik}^* = \max\left(0, u_{ik}^* - \frac{1}{2\Big(\sum_{m=1}^{n}(u_{mk}^*)^2 + D_{ik}^*\Big)}\frac{\partial f(U^*)}{\partial u_{ik}^*}\right). \tag{35}$$

If $u_{ik}^* = 0$, from equation (35), $\frac{\partial f(U^*)}{\partial u_{ik}^*} \geq 0$ is true, and thus KKT optimality conditions (3) for u_{ik}^* are satisfied.

If $u_{ik}^* > 0$, from equation (35), we have

$$u_{ik}^* = u_{ik}^* - \frac{1}{2\Big(\sum_{m=1}^{n}(u_{mk}^*)^2 + D_{ik}^*\Big)}\frac{\partial f(U^*)}{\partial u_{ik}^*},$$

which means that $\frac{\partial f(U^*)}{\partial u_{ik}^*} = 0$, and thus KKT optimality conditions (3) for u_{ik}^*, are satisfied again. Therefore, in summary, U^* satisfies KKT optimality conditions (3).

4. Experiments

Our proposed algorithm (CASNMF) has been theoretically proven to converge fast in terms of the number of iterations with the largest possible search step size at every iteration and can converge with zero initial values and general symmetrical nonnegative matrix A. In this section we carry

out numerical experiments on three datasets to illustrate its performance in practices. CASNMF is compared with two other best algorithms: He's algorithm with $\alpha = 0.99$ and Ding's algorithm with $\beta = 0.99$. All these three competing algorithms are coded in MATLAB R2013a and their codes can be available from the authors on request. The algorithms are judged to converge if the following criterion is met:

$$\frac{\left|OBJ^t - OBJ^{t-1}\right|}{OBJ^t} \leq \varepsilon,$$

where OBJ^t is the value of objective function (1) at iteration step t, and ε is a small positive number and is set as 10^{-6} in this study. To avoid too long running time, the program stops when the pre-defined maximum number of iterations (which is 2000 in this study) is reached. These algorithms are run on a desktop with specifications: a processor of Intel (R)Core (TM) i7-4770 CPU @ 3.40GHZ 3.40GHZ, a RAM of 8.00GB(7.8GB usable), and a 64 bit operating system of Windows 7 Enterprise. In our experiments, all three programs are run with two synthetic networks and one real-life network data. Furthermore, as the initialization of NMF could be critical issues in some applications because of its local convergence [39], especially with initial zero values, our experiments take initial values of U in the two following cases:

Case I: set $u_{ik} = |N(0,1)|, \forall i,k$,

Case II: set $u_{ik} = |N(0,1)|, \forall i,k$, and then randomly set 30% of elements in U as zeros,

where $N(0,1)$ stands for the standard normal distribution. Case I is to set the nonzero initial values while Case II is to set 30% of initial values as zeros. In the following experiments, the same sets of initial values are generated from Case I or Case II for all three competing algorithms when they are compared on the three network datasets.

Synthetic Dataset 1 is created as the adjacent matrix of an undirected network which consists of six separated cliques with sizes of 20, 20, 25, 25, 30, and 30. All diagonal elements of the matrix are zeros. We applied all three competing algorithms to the adjacent matrix of this network with 100 different initial values in each of two cases for $r = 6$, respectively. The results are recorded in Table 1, which shows the number of convergences to different values of objective function (1). Specifically, for all three algorithms the value of objective function (1): c1) diverges (not converge); c2) converges to

72.00 (the smallest minimum value of objective function for this dataset in all experiments); c3) converges to 252.50, or c4) converges to 433.00.

Table 1: Results of Synthetic dataset 1

	He's algorithm with $\alpha = 0.99$	Ding's algorithm with $\beta = 0.99$	our algorithm: CASNMF
initial values of Case I			
# convergences to OBJ.72 (average iterations)	19 (503)	31 (486)	**43** (32)
# convergences to OBJ.252.5 (average iterations)	17 (483)	29 (452)	**57** (98)
# convergences to OBJ.433 (average iterations)	1 (470)	1 (460)	**0**
# Divergences	63	39	**0**
initial values of Case II			
# convergences to OBJ.72 (average iterations)	NA	NA	**66** (30)
# convergences to OBJ.252.5 (average iterations)	NA	NA	**34** (92)
# convergences to OBJ.433	100	100	**0**

From Table 1, one can observe that CASNMF converges for all 100 runs with different initial values of Case I while He's algorithm and Ding's algorithm diverge for 63 and 39 runs, respectively. CASNMF converges for all 100 runs with different initial values of Case II while He's algorithm and Ding's algorithm diverge for all runs. These results illustrate that CASNMF outperforms He's algorithm and Ding's algorithms in term of convergences and robustness to the initial values. Furthermore, the number of runs with

convergence to the smallest minimum value (72.00) of objective function (1) of CASNMF is much more than those of He's algorithm and Ding's algorithm. In addition, from Table 1 we can see that in cases that all algorithms converged to the same values of the objective function (1), CASNMF converged with much less number of iterations than both He's algorithm and Ding's algorithm. These results further illustrate that CASNMF has greater capability to converge to the optimal value of objective function (1) than both He's algorithm and Ding's algorithm.

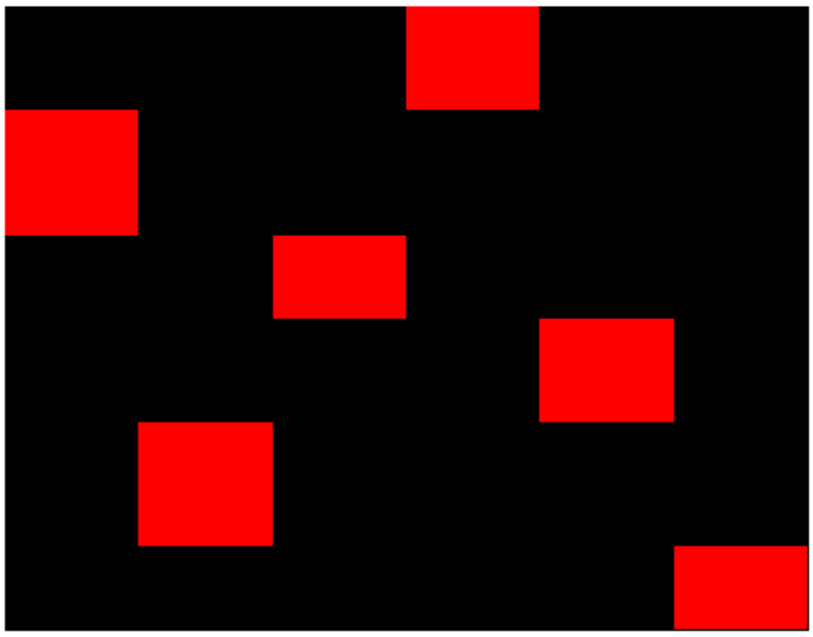

Figure 1: The heatmap of the optimal matrix U with the objective function value of 72 for **Synthetic Dataset 1**.

According to the principle of SYN-NMF [12, 21], ideally the largest components in each column of U should correspond to a dense community in a network [35]. The heatmap of the optimal matrix U with the objective function value of 72 is depicted in Figure 1. From this figure, one can see that each column of matrix U represents a clique and all six cliques have no overlap, which is the real situation of this synthetic dataset. From the matrix U with case OBJ.252.5, the similar heatmap can be obtained. However, from the matrix U with case OBJ.433, the heatmap cannot reflect the six true cliques. After carefully checking, we find that the solution series for both

cases OBJ.72 and OBJ.252.5 converge to a stationary point while the solution series for case OBJ.433 does not converge to a stationary point although it converges to a limit point. From Table 1, CASNMF always converges to a stationary point while other two competing methods do not. Therefore, we can conclude that CASNMF is a powerful and robust algorithm to detect dense communities in a network.

Synthetic dataset 2 is created as $a_{ij} = |N(0,1)|$ with $n = 500$. To fairly compare them, all three competing algorithms are run on this dataset with $r = 30$, starting with the same sets of 50 different initial values of Cases I and II, respectively. The results are recorded in Table 2.

Table 2: Results of Synthetic dataset 2

	He's algorithm with $\alpha = 0.99$	Ding's algorithm with $\beta = 0.99$	our algorithm: CASNMF
initial values of Case I			
Minimum OBJ	1665.70	1665.70	**1658.50**
Maximum OBJ	1705.90	1716.90	**1660.50**
Average OBJ	1672.72	1670.64	**1658.36**
STDEV OBJ	13.93	13.38	**0.49**
initial values of Case II			
Minimum OBJ	1852.70	1822.50	**1658.30**
Maximum OBJ	1846.60	1856.90	**1660.50**
Average OBJ	1835.36	1837.82	**1659.24**
STDEV OBJ	5.60	9.20	**0.43**

From Table 2, we can observe that all three competing algorithms converge for all different initial values of both Cases I and II. Over 50 different initial values of Case I, all three competing algorithms converge; however, the average optimal value of objective function of CASNMF is 1658.36 while those of He's algorithm and Ding's algorithm are 1672.72 and 1670.64, respectively. Furthermore, the standard deviation of CASNMF is 0.49 while those of He's algorithm and Ding's algorithm are 13.93 and 13.38, respectively. These results illustrated that CASNMF more robustly converges to the optimal value of objective function (1) than both He's algorithm and Ding's algorithm. In addition, the minimum optimal value of objective function of CASNMF over 50 runs is smaller than those of He's algorithm and Ding's algorithm while the maximum optimal value of objective function of

CASNMF is even less than the minimum optimal value of objective function of He's algorithm and Ding's algorithm over 50 runs. These results illustrate that CASNMF outperforms He's algorithm and Ding's algorithm in terms of finding the minimum optimal value of objective function (1).

For initial values of Case II, although all three competing algorithms converge again, He's algorithm and Ding's algorithm converge to much larger values of objective function (1) than CASNMF. Furthermore, the results of CASNMF in Case I and Case II have no significant difference while the results of He's algorithm and Ding's algorithm are significantly different. Especially, the average, minimum and maximum optimal values of objective function (1) in Case II are much larger than them in Case I for both He's algorithm and Ding's algorithm, which indicates that in Case II He's algorithm and Ding's algorithm may actually have stopped before the optimal value of objective function (1) is reached.

Real-life dataset is the adjacent matrix of protein-protein interaction network which was applied to identify protein complexes [40, 41]. In this network, there are 5014 proteins and 44695 interactions. Thus, its adjacent matrix is a 5014×5014 symmetric nonnegative matrix with 44695 elements being 1 and others being 0. All three competing algorithms are run on this large and sparse adjacent matrix with various values of $r(= 10, 20, 30, 50, 100)$, starting with the same sets of 10 different initial values of Cases I and II, respectively. Table 3 records the results with $r = 50$.

Table 3: Results of PPI Network with r=50

	He's algorithm with $\alpha = 0.99$	Ding's algorithm with $\beta = 0.99$	our algorithm: CASNMF
initial values of Case I			
Average OBJ	NA	17822.75	**17769.4**
STDEV OBJ	NA	24.12	**11.02**
Diverged	10	2	**0**
initial values of Case II			
Average OBJ	NA	18804.2	**17762.8**
STDEV OBJ	NA	31.12	**7.61**
Diverged	10	5	**0**

From Table 3, we can see that our algorithm CASNMF converges with all 20 different initial values while He's algorithm diverges with them and

Ding's algorithm converges with some initial values while diverging with other initial values. Although Ding's algorithm can converge, it converges to a much larger value of objective function (1) than CASNMF. Furthermore, CANSNMF is more robust than Ding's algorithm in terms of standard deviation of converged values of objective function. In contrast, CASNMF converges to the steady values of objective functions with initial values of both cases. Although the results with other values of $r(= 10, 20, 30, 100)$ are not presented in this paper, the same conclusions can be drawn as those with $r = 50$, too.

5. Conclusions

In this paper, we have studied SYM-NMF which is formulated as the optimization problem (1). After analyzing the existing algorithms we have proposed a converged algorithm (CASNMF) for solving this problem. Based on optimization principle and local auxiliary function method, we have proved that our algorithm converges to a stationary point even if matrix A is not positive definite and elements of initial U can be zero. The experimental results on three datasets have also verified our theoretical results. From computational experiments on three datasets with different initial values, our proposed CASNMF always converges to a stationary point while He's algorithm and Ding's algorithm often diverge or converge to a non-stationary point. In addition, our proposed CASNMF converges to a stationary point with a smaller objective function value than He's algorithm and Ding's algorithm when they converge. One direction of our future work is to apply the CASNMF to solve problems in complex network analysis.

Acknowledgments

This work is supported in part by Natural Science and Engineering Research Council of Canada (NSERC), China Scholarship Council (CSC) and by the National Natural Science Foundation of China under Grant No. 61571052.

Appendix A.

In this Appendix, we derive the derivatives of the objective function (1) with respect to an element u_{ik} of matrix U, which are used in the proof

of Theorems 2 and 4. The objective function (1) can be expressed by the elements of U as follows:

$$\begin{aligned} f(U) &= \frac{1}{2}\left\|A - UU^T\right\|_F^2 = \frac{1}{2}\sum_{m=1}^{n}\sum_{l=1}^{n}(a_{ml} - \sum_{s=1}^{r} u_{ms}u_{ls})^2 \\ &= \frac{1}{2}\sum_{m=1}^{n}\sum_{l=1}^{n}\left[a_{ml}^2 - 2a_{ml}\sum_{s=1}^{r} u_{ms}u_{ls} + (\sum_{s=1}^{r} u_{ms}u_{ls})^2\right]. \end{aligned} \tag{A.1}$$

Taking the first derivative of the objective function (1) with respect to u_{ik} yields to

$$\begin{aligned} \frac{\partial f(U)}{\partial u_{ik}} &= \frac{1}{2}\sum_{m=1}^{n}\sum_{l=1}^{n}\Bigg[- 2a_{ml}\sum_{s=1}^{r}(\delta_{ms}^{ik}u_{ls} + u_{ms}\delta_{ls}^{ik}) \\ &\qquad +2(\sum_{s=1}^{r} u_{ms}u_{ls})\sum_{s=1}^{r}(\delta_{ms}^{ik}u_{ls} + u_{ms}\delta_{ls}^{ik})\Bigg] \\ &= -\sum_{l=1}^{n} a_{il}u_{lk} - \sum_{m=1}^{n} a_{mi}u_{mk} + \sum_{l=1}^{n}\sum_{s=1}^{r} u_{is}u_{ls}u_{lk} + \sum_{m=1}^{n}\sum_{s=1}^{r} u_{ms}u_{is}u_{mk} \\ &= 2\sum_{m=1}^{n}(\sum_{s=1}^{r}(u_{ms}u_{is}) - a_{mi})u_{mk}, \end{aligned} \tag{A.2}$$

where $\delta_{ms}^{ik} = 1$ if $i = m$ and $k = s$, and 0 otherwise. The second derivative of the objective function (1) with respect to u_{ik} can be obtained as follows.

$$\begin{aligned} \frac{\partial^2 f(U)}{\partial (u_{ik})^2} &= \frac{\partial}{\partial u_{ik}}\Big[2\sum_{m=1}^{n}(\sum_{s=1}^{r}(u_{ms}u_{is}) - a_{mi})u_{mk}\Big] \\ &= 2\sum_{m=1}^{n}\left(\sum_{s=1}^{r}(\delta_{ms}^{ik}u_{is} + u_{ms}\delta_{is}^{ik})\right)u_{mk} \\ &\quad + 2\sum_{m=1}^{n}\left(\sum_{s=1}^{r}(u_{ms}u_{is}) - a_{mi}\right)\delta_{mk}^{ik} \\ &= 2\Big[u_{ik}^2 + \sum_{m=1}^{n} u_{mk}^2 + \sum_{s=1}^{r} u_{is}^2 - a_{ii}\Big]. \end{aligned} \tag{A.3}$$

Furthermore, the third and fourth derivatives of the objective function (1) with respect to u_{ik} are

$$\frac{\partial^3 f(U)}{\partial (u_{ik})^3} = \frac{\partial}{\partial u_{ik}} 2\Big[u_{ik}^2 + \sum_{m=1}^{n} u_{mk}^2 + \sum_{s=1}^{r} u_{is}^2 - a_{ii}\Big] = 12u_{ik}, \tag{A.4}$$

$$\frac{\partial^4 f(U)}{\partial (u_{ik})^4} = 12. \tag{A.5}$$

From the above, the second derivatives are still function of elements of U while the third and fourth derivatives are non zeros, which are different from the situation in the general NMF.

Appendix B.

In this Appendix, we present a Theorem and its proof. This Theorem is used in the proof of Theorem 3.

Theorem: Let $U \in R^{n\times r}$. Then we have that

$$\big\|U\big\|_F^2 \leq \sqrt{r}\big\|UU^T\big\|_F \tag{B.1}$$

Proof. By the definition of Frobenius norm, we have that

$$\big\|U\big\|_F^2 = Trace(UU^T) = \sum_{s=1}^{r} \sigma_s^2 \tag{B.2}$$

$$\big\|UU^T\big\|_F^2 = Trace(UU^TUU^T) = \sum_{s=1}^{r} \sigma_s^4 \tag{B.3}$$

where $\sigma_s (s = 1, 2, \ldots, r)$ is the singular value of U. By applying Cauchy–Schwarz inequality to (B.2), we have

$$\begin{aligned}\big\|U\big\|_F^2 = \sum_{s=1}^{r} \sigma_s^2 = \sum_{s=1}^{r} (1^* \sigma_s^2) &\leq \sqrt{\Big(\sum_{s=1}^{r} 1^2\Big)\Big(\sum_{s=1}^{r} (\sigma_s^2)^2\Big)} \\ &= \sqrt{r}\sqrt{\Big(\sum_{s=1}^{r} \sigma_s^4\Big)}\end{aligned}$$

Substituting (B.3) to the above equation yields to (B.1).

Reference

[1] H. Hotelling, Analysis of a complex of statistical variables into principal components, Journal of Educational Psychology 24 (6) (1933) 417.

[2] C. M. Bishop, Pattern recognition and machine learning, Springer, New York, 2007.

[3] A. Hyvarinen, Fast and robust fixed-point algorithms for independent component analysis, IEEE transactions on Neural Networks 10 (3) (1999) 626–634.

[4] A. Hyvärinen, J. Karhunen, E. Oja, Independent component analysis, John Wiley & Sons, New York, 2001.

[5] J. C. Liao, R. Boscolo, Y.-L. Yang, L. M. Tran, C. Sabatti, V. P. Roychowdhury, Network component analysis: reconstruction of regulatory signals in biological systems, Proceedings of the National Academy of Sciences 100 (26) (2003) 15522–15527.

[6] R. Boscolo, C. Sabatti, J. C. Liao, V. P. Roychowdhury, A generalized framework for network component analysis, IEEE/ACM Transactions on Computational Biology and Bioinformatics 2 (4) (2005) 289–301.

[7] M. Petrou, C. Petrou, Image processing: the fundamentals, John Wiley & Sons, New York, 2010.

[8] A. N. Srivastava, M. Sahami, Text mining: Classification, clustering, and applications, CRC Press, Boca Raton, Fla, 2009.

[9] C. Stark, B.-J. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, M. Tyers, Biogrid: a general repository for interaction datasets, Nucleic Acids Research 34 (suppl 1) (2006) D535–D539.

[10] D. D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.

[11] L.-P. Tian, L. Liu, F.-X. Wu, Matrix decomposition methods in bioinformatics, Current Bioinformatics 8 (2) (2013) 259–266.

[12] C. H. Ding, X. He, H. D. Simon, On the equivalence of nonnegative matrix factorization and spectral clustering, in: SDM, Vol. 5, SIAM, 2005, pp. 606–610.

[13] Y. Chen, M. Rege, M. Dong, J. Hua, Non-negative matrix factorization for semi-supervised data clustering, Knowledge and Information Systems 17 (3) (2008) 355–379.

[14] Y. Chen, L. Wang, M. Dong, Non-negative matrix factorization for semisupervised heterogeneous data coclustering, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1459–1474.

[15] N. Del Buono, G. Pio, Non-negative matrix tri-factorization for co-clustering: An analysis of the block matrix, Information Sciences 301 (2015) 13–26.

[16] Y. Lu, Z. Lai, Y. Xu, J. You, X. Li, C. Yuan, Projective robust nonnegative factorization, Information Sciences 364 (2016) 16–32.

[17] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, R. J. Plemmons, Algorithms and applications for approximate nonnegative matrix factorization, Computational Statistics & Data Analysis 52 (1) (2007) 155–173.

[18] P. O. Hoyer, Non-negative matrix factorization with sparseness constraints, Journal of Machine Learning Research 5 (2004) 1457–1469.

[19] F. Shahnaz, M. W. Berry, V. P. Pauca, R. J. Plemmons, Document clustering using nonnegative matrix factorization, Information Processing & Management 42 (2) (2006) 373–386.

[20] W. Xu, X. Liu, Y. Gong, Document clustering based on non-negative matrix factorization, in: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, ACM, 2003, pp. 267–273.

[21] D. Wang, T. Li, S. Zhu, C. Ding, Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization, in: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2008, pp. 307–314.

[22] H. Kim, H. Park, Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis, Bioinformatics 23 (12) (2007) 1495–1502.

[23] D. Greene, G. Cagney, N. Krogan, P. Cunningham, Ensemble non-negative matrix factorization methods for clustering protein–protein interactions, Bioinformatics 24 (15) (2008) 1722–1728.

[24] Y. Li, A. Ngom, Nonnegative least-squares methods for the classification of high-dimensional biological data, IEEE/ACM Transactions on Computational Biology and Bioinformatics 10 (2) (2013) 447–456.

[25] Q. Qi, Y. Zhao, M. Li, R. Simon, Non-negative matrix factorization of gene expression profiles: a plug-in for brb-arraytools, Bioinformatics 25 (4) (2009) 545–547.

[26] D. D. Lee, H. S. Seung, Algorithms for non-negative matrix factorization, in: Advances in Neural Information Processing Systems, 2001, pp. 556–562.

[27] C.-J. Lin, On the convergence of multiplicative update algorithms for nonnegative matrix factorization, IEEE Transactions on Neural Networks 18 (6) (2007) 1589–1596.

[28] C.-J. Lin, Projected gradient methods for nonnegative matrix factorization, Neural Computation 19 (10) (2007) 2756–2779.

[29] L.-X. Li, L. Wu, H.-S. Zhang, F.-X. Wu, A fast algorithm for nonnegative matrix factorization and its convergence, IEEE Transactions on Neural Networks and Learning Systems 25 (10) (2014) 1855–1863.

[30] Y.-X. Wang, Y.-J. Zhang, Nonnegative matrix factorization: A comprehensive review, IEEE Transactions on Knowledge and Data Engineering 25 (6) (2013) 1336–1353.

[31] R. Zass, A. Shashua, A unifying approach to hard and probabilistic clustering, in: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, Vol. 1, IEEE, 2005, pp. 294–301.

[32] B. Long, Z. M. Zhang, P. S. Yu, Co-clustering by block value decomposition, in: Proceedings of the Eleventh ACM SIGKDD International

Conference on Knowledge Discovery in Data Mining, ACM, 2005, pp. 635–640.

[33] B. Long, Z. M. Zhang, X. Wu, P. S. Yu, Relational clustering by symmetric convex coding, in: Proceedings of the 24th International Conference on Machine Learning, ACM, 2007, pp. 569–576.

[34] Z. He, S. Xie, R. Zdunek, G. Zhou, A. Cichocki, Symmetric nonnegative matrix factorization: Algorithms and applications to probabilistic clustering, IEEE Transactions on Neural Networks 22 (12) (2011) 2117–2131.

[35] X. Cao, X. Wang, D. Jin, Y. Cao, D. He, Identifying overlapping communities as well as hubs and outliers via nonnegative matrix factorization, Scientific Reports 3 (2013) 2993.

[36] B. Chen, W. Fan, J. Liu, F.-X. Wu, Identifying protein complexes and functional modulesfrom static ppi networks to dynamic ppi networks, Briefings in Bioinformatics 15 (2) (2014) 177–194.

[37] L.-Z. Liu, F.-X. Wu, W.-J. Zhang, Estimating parameters of s-systems by an auxiliary function guided coordinate descent method, Systems Science & Control Engineering: An Open Access Journal 2 (1) (2014) 125–134.

[38] S. Boyd, L. Vandenberghe, Convex optimization, Cambridge University Press, Cambridge, UK, 2004.

[39] G. Casalino, N. Del Buono, C. Mencar, Subtractive clustering for seeding non-negative matrix factorizations, Information Sciences 257 (2014) 369–387.

[40] B. Chen, J. Shi, S. Zhang, F.-X. Wu, Identifying protein complexes in protein–protein interaction networks by using clique seeds and graph entropy, Proteomics 13 (2) (2013) 269–277.

[41] B. Chen, F.-X. Wu, Identifying protein complexes based on multiple topological structures in ppi networks, IEEE Transactions on Nanobioscience 12 (3) (2013) 165–172.

Li–Ping Tian received the M. Sc. degree in Applied Mathematics from Dalian University of Technology, Dalian, China, in 1993. He is currently a professor of School of Information, Beijing Wuzi University. His current research interests include molecular systems biology, biological system identification and parameter estimation, inverse problem and stability of dynamic systems. Professor Tian has published more than 60 technical papers in refereed journals and conference proceedings. He is serving as the editorial member of two journals, and also reviewing papers for several journals.

Ping Luo received his B.Sc. degree in computer science and technology from Hunan University, China, in 2010 and M.Sc. degree in biomedical engineering from Beijing Institute of Technology, China, 1n 2015. Currently, he is working toward the Ph.D. degree in the Division of Biomedical Engineering at the University of Saskatchewan, Saskatoon, Canada. His current research interests include analysis of large-scale biological data, modeling, analysis, and control of molecular biological networks.

Haiying Wang received his PhD degree on artificial intelligence in biomedicine in 2004 and he is currently a Senior Lecturer in the School of Computing and Mathematics at Ulster University, Belfast, UK. His research interests lie broadly within the areas of artificial intelligence, complex network analysis, computational biology and bioinformatics. He has a particular research interest and expertise in the development of bio-inspired, self-adaptive and self-organization systems, advanced techniques for the extraction of functional modules from complex systems, and network-based approaches to the field of systems biology. Since 2004, he has published more than 110 peer-reviewed papers in refereed journals and conference proceedings.

Huiru Zheng received her PhD degree on data mining and bioinformatics from the University of Ulster, UK in 2003. Her re-search area lies on the broad area of healthcare informatics, including bioinformatics, medical informatics, data mining and artificial intelligence and their applications on systems biology, telecare and tele-medicine. She has published over 180 research papers in peer reviewed international journals and conferences. Dr. Zheng is currently a Reader with the School of Computing and Mathematics at the University of Ulster.

Fang–Xiang Wu is currently a full professor of the Department of Computer Science, the Department of Mechanical Engineering and the Division of Biomedical Engineering at the University of Saskatchewan. His current research interests include Machine learning in bioinformatics, Biological data analytics, Complex bionetwork analytics, Network controllability, and Nonlinear biodynamic analytics. Dr. Wu has published more than 260 technical papers in refereed journals and conference proceedings. Dr. Wu is serving as the editorial board member of five international journals and as the guest editor of several international journals, and as the program committee chair or member of several international conferences.