GAUSS-SEIDEL BASED NON-NEGATIVE MATRIX FACTORIZATION FOR GENE EXPRESSION CLUSTERING

Qing Liao[†]*, Naiyang Guan*[‡]*, Qian Zhang*[†]

[†]Dept. of Computer Science & Engineering, The Hong Kong University of Science and Technology
[‡]School of Computer Science, National University of Defense Technology
qnature@ust.hk, ny_guan@nudt.edu.cn, qianzh@cse.ust.hk

ABSTRACT

Genome-wide expression data consists of millions of measurements towards large number of genes, and thus it is challenging for human beings to directly analyze such large-scale data. Clustering provides a more convenient way to analyze gene expression data because it can subdivide raw data into comprehensive classes. However, the number of probed genes is rather greater than the number of samples, and this makes conventional clustering methods perform unsatisfactorily. In this paper, we propose a Gauss-Seidel based non-negative matrix factorization (GSNMF) method to overcome such imbalance deficiency between features and samples. In particular, GSNMF iteratively projects gene expression data onto the learned subspace followed by adaptively updating the cluster centroids based on the projected data. Since this data projection strategy significantly reduces the influence of imbalance between the number of samples and the number of genes, GSNMF performs better than traditional clustering methods in gene expression clustering. Since GSNMF updates each factor matrix by solution of a linear system obtained by the Gauss-Seidel method, it converges rapidly without neither complex line search nor matrix inverse operators. Experimental results on several cancer expression datasets confirm both efficiency and effectiveness of GSNMF comparing with the representative NMF methods and conventional clustering methods.

Index Terms— Gene expression clustering, non-negative matrix factorization, Gauss-Seidel method.

1. INTRODUCTION

Microarray technologies have made it possible to monitor the expression levels of tens of thousands genes in parallel. A microarray typically assesses a large number of DNA sequences, e.g., genes, cDNA clones, or expressed sequence tags under various conditions [1], e.g., different time series, collections of different tissue samples, different states when stimuli is on and off, and several different stimulus. As the progress of Genomic research, Human being's ability of gathering genome-wide expression data has far outstripped the ability of human beings to analyze the gene expression data. To meet the requirement of analyzing genome-wide gene expression data, life scientists usually distill gene expression data down to a more comprehensible level by using clustering tools. In general, gene expression clustering subdivides a set of genes or samples into several subsets so that genes in an identical subset have similar gene expression pattern and unearths similar bio-process, gene function, gene regulation, and subtypes of cells [2].

Many clustering methods such as hierarchical clustering [3][4], K-means [5][6], self-organizing maps (SOM, [7][8][9]), and non-negative matrix factorization (NMF) [10][11][12][13] have been applied to clustering gene expression data. Among them, NMF [14][15] is one of the most powerful method and Brunet *et al.* [11] shows that NMF has better performance than hierarchical clustering and SOM. Kim and Park [12] illustrated that sparse NMF can achieve satisfactory clustering performance. Moreover, Zheng *et al.* [13] employed independent component analysis (ICA) to select a subset of genes to further improve the clustering performance of NMF.

Although NMF have achieved great successes in gene expression clustering, it suffers from the deficiency of serious imbalance between the number of genes and the number of samples. Generally speaking, a typical microarray data contains thousands of genes on each chip, and the number of collected samples is usually no more than 100. Such imbalance deficiency [16] between large number of genes and small number of samples makes most clustering methods to perform unsatisfactorily, because it is difficult to choose primary genes based only on few samples.

In this paper, we proposed a Gauss-Seidel based NMF (GSNMF) to overcome this deficiency. Similar to traditional NMF, GSNMF iteratively optimizes the cluster centroids and the coefficients until convergence. Distinguished from traditional NMF, for optimizing the coefficients, GSNMF projects the data onto a data-driven subspace, and updates the coefficient matrix with the solution of the linear system obtained by using the Gauss-Seidel method. The cluster centroids are optimized in the same way. Since the utilized Gauss-Seidel method [17][18] needs neither complex line search nor matrix inverse operator, GSNMF is more efficient than other NMF solvers such as the popular multiplicative update rule (MUR) [15] and projected gradient method [19]. In each iteration round, GSNMF normalizes both factor matrices, and this normalization makes the objective function of GSNMF to be upper-bounded by that of NMF. Therefore, GSNMF can obtain a local minimum of NMF and can overcome the deficiency of imbalance between the number of genes and the number of samples. Experimental results on several cancer expression datasets confirm both efficiency and effectiveness of GSNMF in gene expression clustering.

2. GAUSS-SEIDEL BASED NON-NEGATIVE MATRIX FACTORIZATION

Non-negative matrix factorization (NMF, [14]) learns two low-dimensional non-negative matrices, i.e., W and H, to approximate a high-dimensional non-negative matrix V, by minimizing the distance between V and WH, namely $D(V|WH)$. NMF is usually optimized by using multiplicative update rule (MUR) [15]. For example, if the distance D is measured by the Frobenius norm, the MUR is

$$\begin{cases} W \leftarrow W \otimes \frac{VH^T}{WHH^T} \\ H \leftarrow H \otimes \frac{W^TV}{W^TWH} \end{cases}. \quad (1)$$

However, MUR converges slowly because it is intrinsically a first-order gradient based method [20], and cannot produce zero entries

because zero entry does not change in the subsequent iterations. To overcome these deficiencies, Guan *et al*. [21] proposed an NeNMF method to alternating optimize both W and H by solving the non-negative least squares (NNLS) problem with Nesterov's method. Although traditional NMF methods perform well on some tasks such as image analysis and text mining, they cannot perform satisfactorily on gene expression clustering because the number of genes is usually much larger than the number of samples.

To remedy this problem, we propose a novel Gauss-Seidel based NMF (GSNMF) method to alternating update each matrix by using the Gauss-Seidel method. In particular, GSNMF transforms the subproblem of optimizing each matrix to solving a linear system. The total procedure is summarized in **Algorithm 1**. In the second statement, GSNMF updates both matrices by solving linear systems by using the Gauss-Seidel method [17][18].

2.1. An Illustrative Example

Before introducing the GS algorithm, we present a simple example to illustrate its procedure.

Problem: Given $V = \begin{bmatrix} 8 & 9 \\ 9 & 6 \\ 1 & 1 \end{bmatrix}$ and $A = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \end{bmatrix}$, find $H = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{bmatrix}$ such that $V = AH$.

Solution: Firstly, we randomly initialize the matrix H, e.g., $H^1 = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 5 & 3 \end{bmatrix}$, and divide A into three components including a upper triangular matrix, a lower triangular matrix, and a diagonal matrix, i.e.,

$$\begin{aligned} A &= U + D + U^T \\ &= \begin{bmatrix} 0 & 2 & 2 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 2 & 2 & 2 \end{bmatrix}. \end{aligned} \quad (2)$$

By multiplying the equation (2) from the right hand by H, we could equally obtain the following equation: $AH = UH + DH + U^T H$. By simple algebra, the equation $V = AH$ is equivalent to $V - DH = UH + U^T H$. Given H^k, we can obtain H^{k+1} by solving the following linear system: $V - UH^k = DH^{k+1} + U^T H^{k+1}$.

For example, from H^1, we can obtain H^2 as follows:

$$\begin{bmatrix} 8 & 9 \\ 9 & 6 \\ 1 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 2 & 2 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 5 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 2 & 2 & 0 \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \\ h_{31} & h_{32} \end{bmatrix}$$

According to the Gauss-Seidel (GS) method [17][18], we can easily compute H row by row. At the first step, we have $[h_{11}, h_{12}] = [-6, -7]$. Continuous GS steps output $[h_{21}, h_{22}] = [11, 14]$ and $[h_{31}, h_{32}] = [-9, -13]$. Finally, we get $H^2 = \begin{bmatrix} -6 & -7 \\ 11 & 14 \\ -9 & -13 \end{bmatrix}$.

Note that different initializations of H result in different minimizers, however, initialization does not seriously influence the minimum because the problem is convex.

2.2. The GS Algorithm

Inspired by the illustrative example, we employ the GS method to optimize both factor matrices, i.e., W and H, in NMF. Take the optimization of H for example. It is well-known that the GS method is designed to find the solution of model $b \approx Ax$, where b is an target vector of size n and A is an observation matrix, and the observation matrix A is constrained to be symmetric. The GS method cannot be directly applied to solve the linear system $V = WH$ as W is unnecessarily to be symmetric. To remedy this problem, we transform the subproblem of optimizing H to an approximate problem, i.e., $W^T V = W^T WH$. Since $W^T W$ is a symmetric matrix, the GS method can be naturally applied to solve this linear system. Similarly, when H is fixed, we could calculate W by the linear system $VH^T = WHH^T$ or $V^T = HH^T W^T$. Here we focus on the optimization of H as the optimization of W follows the same way. For the convenience of derivatives, we denote $W^T V$ and $W^T W$ as B and A, respectively.

Algorithm 1 Gauss-Seidel Based Non-negative Matrix Factorization

Input: $V \in R_+^{m \times n}$, $1 \le r \le \min\{m, n\}$.
Output: $W \in R_+^{m \times r}$, $H \in R_+^{r \times n}$.
1: Initialize: $W^1 \ge 0, H^1 \ge 0, k = 1$.
2: **Repeat**
$H^{k+1} = GS\left((\mathrm{W}^k)^T \mathrm{W}^k, (\mathrm{W}^k)^T \mathrm{V}, \mathrm{H}^k, tol(H^k)\right)$.
$W^{k+1} = GS\left(H^{k+1}(H^{k+1})^T, H^{k+1}V^T, (W^k)^T, tol(W^k)\right)$.
$W^{k+1} = (W^{k+1})^T$.
$k \leftarrow k + 1$.
3: **Until** {Stopping criterion is satisfied}.
4: $W = W^k$, $H = H^k$.

In particular, the GS method divides A into two parts, i.e., $A = L + U$, where L is a lower triangular matrix and U is a strictly upper triangular matrix. Substituting $A = L + U$ into the model $B = AH$, we have a new linear system

$$B - UH = LH. \quad (3)$$

Based on the equation (3), we can get the left-hand side of H^{k+1} using the previous value of H^k on the right-hand side, i.e.,

$$L^{-1}(V - UH^k) = H^{k+1}, \quad (4)$$

where k is the iteration counter. Since L is a lower triangular matrix, the GS method computes the elements of H^{k+1} sequentially by using forward substitution. Specially, we further divide L into $L = U^T + D$ since A is symmetric matrix, where $D = diag(A)$. Then the system can be rewritten into:

$$B - UH^k = U^T H^{k+1} + DH^{k+1}. \quad (5)$$

By simple algebra, we have

$$\begin{aligned} [B_1, B_2, ..., B_n] - U\left[H_1^k, H_2^k, ..., H_n^k\right] = \\ U^T\left[H_1^{k+1}, H_2^{k+1}, ..., H_n^{k+1}\right] + D\left[H_1^{k+1}, H_2^{k+1}, ..., H_n^{k+1}\right], \end{aligned} \quad (6)$$

where B_i denotes the i-th column of B, H_i^k denotes the i-th column of H^k.

Looking more carefully at the formulation (5), we have

$$\begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \vdots & \vdots \\ b_{n1} & \cdots & b_{nn} \end{pmatrix} - U \begin{pmatrix} h_{11}^{k+1} & \cdots & h_{1n}^{k+1} \\ \vdots & \vdots & \vdots \\ h_{n1}^{k+1} & \cdots & h_{nn}^{k+1} \end{pmatrix} = \mathrm{U}^T \begin{pmatrix} h_{11}^k & \cdots & h_{1n}^k \\ \vdots & \vdots & \vdots \\ h_{n1}^k & \cdots & h_{nn}^k \end{pmatrix} + \begin{pmatrix} a_{11}h_{11}^{k+1} & \cdots & a_{1n}h_{1n}^k \\ \vdots & \vdots & \vdots \\ a_{n1}h_{n1}^k & \cdots & a_{nn}h_{nn}^k \end{pmatrix}.$$

Table 1. Time complexities of NMF, NeNMF, and GSNMF.

Algorithm	Time complexity of one iteration round
NMF [15]	$O(mnr + mr^2 + nr^2)$
NeNMF [21]	$O(mnr + mr^2 + nr^2) + K \times O(mr^2 + nr^2)$
GSNMF	$O(mnr + mr^2 + nr^2)$

So we know that the row vector satisfies

$$B_i - \sum_{j>i} \left(a_{ij} H_{j:}^k\right) = \sum_{j<i} \left(a_{ji} H_{j:}^{k+1}\right) + a_{ii} H_{j:}^{k+1}. \quad (7)$$

That is to say,

$$(b_{i1}, \cdots, b_{in}) - \left(\sum_{j>i} a_{ij} h_{j1}^k, \cdots, \sum_{j>i} a_{ij} h_{jn}^k\right) = \quad (8)$$

$$\left(\sum_{j<i} a_{ji} h_{j1}^{k+1}, \cdots, \sum_{j<i} a_{ji} h_{jn}^{k+1}\right) + \left(a_{ii} h_{j1}^{k+1}, \cdots, a_{ii} h_{jn}^{k+1}\right),$$

where $B_{i:}$ is the i-th row vector of B, $H_{j:}^k$ is the k-th iteration of the j-th row vector of H. Therefore, $H_{j:}^{k+1}$ could be calculated connivently in the following way:

$$H_{j:}^{k+1} = \frac{1}{a_{ii}} \left(B_{i:} - U_{i:} H_{j:}^k - (U_{:i})^T H_{j:}^{k+1}\right)$$

$$= \frac{1}{a_{ii}} \left(B_{i:} - \sum_{j>i} a_{ij} H_{j:}^k - \sum_{j<i} a_{ji} H_{j:}^{k+1}\right). \quad (9)$$

The GS algorithm iteratively updates H until the stopping criterion [21] is satisfied.

2.3. Computational Complexity

The main time cost of GS method is spent on computing UH in (9). Since U is a strictly upper triangular matrix, the matrix-vector multiplication operators in UH costs $(r-1)r$ multiplications and $(r-2)r$ additions. In summary, the equation (9) costs $(r-1)r$ multiplications, r divisions, and $(r-1)r$ additions. In summary, the time complexity of GS method is $O(r^2)$. Therefore, the time complexity of one iteration for updating W^{k+1} and H^{k+1} is $O(r^2)$. Recalling that the GS method computes $A = W^T W$ and $B = W^T V$ beforehand, the time complexity of GSNMF, i.e., **Algorithm 1**, is $O(mnr + mr^2 + nr^2)$. **Table 1** shows that the complexity of GSNMF is comparable to MUR and NeNMF. In NeNMF algorithm, the K is a number less than r. Howerver, GSNMF converges much faster at each iteration round since the rank r is much smaller than $\min\{m, n\}$. As shown in the experimental result, we can see GSNMF runs much faster than both NeNMF and MUR.

2.4. Discussion

In GSNMF, we actually optimize H by solving a transformed problem, i.e., $\min_{W\geq 0, H\geq 0} \left\|W^T V - W^T W H\right\|_F^2$, instead of solving the original problem $\min_{W\geq 0, H\geq 0} \|V - WH\|_F^2$. According to the definition of the Frobenius norm, we have the following properties:

$$\|AB\|_F \leq \|A\|_F \|B\|_F. \quad (10)$$

Table 2. Summarization of six cancer gene expression datasets.

Dataset	n	m	r
Gastric cancer (GSE2685)	30	4522	2
Gastric cancer 2 (GSE2685)	30	4522	3
Lymphoma & leukemia (GSE1577)	29	15434	3
Lymphoma & leukemia 2 (GSE1577)	19	15434	2
Airway epithelium database (GSE5060)	22	18651	4
Soft tissue sarcomas (GSE2719)	39	18753	8

n: #(samples), m: #(genes), and r:#(cancer classes)

So, we have $\left\|W^T V - W^T W H\right\|_F^2 \leq \|W\|_F^2 \|V - WH\|_F^2$. If we normalize the matrix W before each call of the GS method and update H accordingly without influencing the approximation WH, the Frobenius norm of W can be made constant. This implies that the difference between errors in the transformed problem $W^T V \approx W^T W H$ and the original problem $V \approx WH$ is bounded.

Since GSNMF transforms the original non-negative least squares (NNLS, [21]) problem with scale $(m \times r \times n)$ into an linear system with scale $(r \times r \times n)$, GSNMF reduces the risk of failure cluster assignments caused by the imbalance between numbers of genes and samples in gene expression clustering.

3. EXPERIMENTS

In this experiments, we validate the effectiveness of GSNMF in cancer gene expression clustering comparing with NMF [14] and NeNMF [21]. Since NMF is non-convex with respect to both W and H, to avoid the influence of random initialization, we performed 100 independent trials, and evaluated the clustering performance by average accuracy and mutual information.

Datasets. We validated GSNMF on six popular cancer gene expression datasets which includes two gastric cancer databases, two lymphoma & leukemia databases, one airway epithelium database and one human soft tissue sarcomas database. Gastric cancer dataset contains 30 samples, 4522 genes with two diagnostic classes: normal/advanced gastric cancer tissues. And gastric cancer2 dataset contains 30 samples, 4522 genes with three diagnostic classes: normal/diffuse and intestinal gastric cancer tissues. Lymphoma & leukemia dataset contains 29 samples, 15434 genes with three diagnostic classes: T-cell lymphoblastic lymphoma (T-LL)/T-cell acute lymphoblastic leukemia (T-ALL) and B-cell acute lymphoblastic leukemia (B-ALL). And lymphoma & leukemia dataset contains 29 samples, 15434 genes with two diagnostic classes: T-cell lymphoblastic lymphoma (T-LL)/ T-cell acute lymphoblastic leukemia (T-ALL). The Airway epithelium database contains 22 samples, 18651 genes with four diagnostic classes: normal nonsmokers/ normal smokers/ smokers with early chronic obstructive lung disease (COPD), and smokers with established COPD. And soft tissue sarcomas contains 39 samples, 18753 genes with eight human tissue sarcomas. Table 2 briefly summarizes their descriptions. The top four rows are orange data[1] analyzed by bio-laboratory and the rest two datasets are GSE data[2]. In the soft tissue sarcomas dataset, we removed all the single specific experiment samples to reduce the noise. The genes without the Present(P) calls in all samples were excluded from the analysis to reduce the amount of noise in the datasets, and if this was the case, we use bold-face to mark the number of genes which lists in Table 2.

Clustering performance. Here we validate the clustering performance of all algorithms in terms of two criterions, i.e., accuracy

[1]The orange datasets are available at http://www.biolab.si/supp/bi-cancer/projections/.

[2]The GSE data can be collected from http://www.ncbi.nlm.nih.gov/gds.

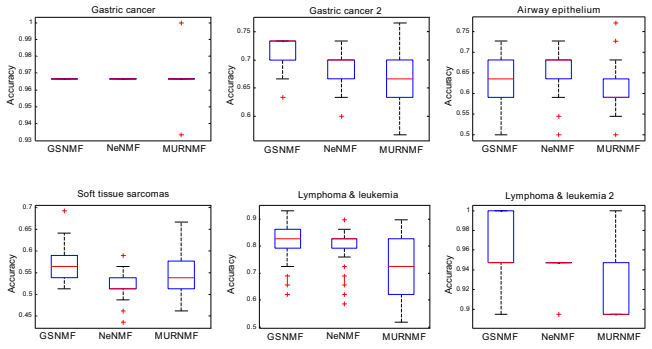

Fig. 1. The box plots of the clustering accuracies for the three NMF algorithms over 100 runs on all the six gene expression datasets: gastric cancer, gastric cancer 2, airway epithelium database, soft tissue sarcomas, lymphoma & leukemia, lymphoma & leukemia 2.

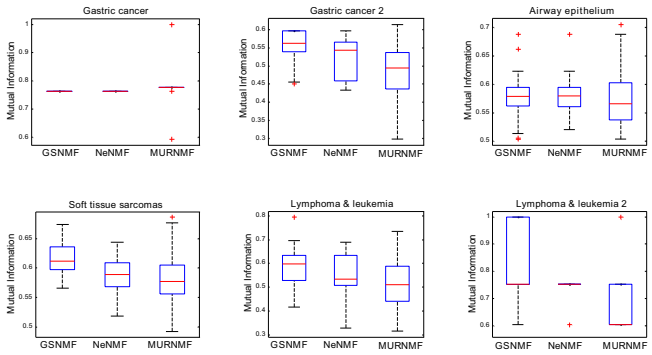

Fig. 2. The box plots of the clustering Mutual Information for the three NMF algorithms over 100 runs on all the six gene expression datasets: gastric cancer, gastric cancer 2, airway epithelium database, soft tissue sarcomas, lymphoma & leukemia, lymphoma & leukemia 2.

(AC) and mutual information (MI). Accuracy estimates the overall cluster performance by using the percentage of correctly clustered samples. Mutual information measures the independency between the predicted clusters and the ground truth, and it equals zero if and only if the predicted clusters are strictly independent from the ground truth.

It can be defined as

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right), \qquad (11)$$

where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distribution functions of X and Y respectively. Mutual information measures the similarity of two different groups. In this paper, we use MI to calculate correlation of predicted label matrix Y and ground truth label matrix X. Generally speaking, if MI equals 1, it means X shares all the information in Y. If MI is 0, X and Y are absolutely different.

We used boxplot to depict the averaged AC and MI for 100 trials. Figure 3 shows the clustering accuracy of the three NMF algorithms on all datasets we used. From the figure we could see that the accuracy of GSNMF is much higher than MUR and is slightly higher than NeNMF. Figure 3 shows the MI of the three NMF algorithms on all datasets. It is obvious that GSNMF achieve the best performance on all datasets.

Figure 3 depicts the convergence curves of three NMF algorithms. Notably, GSNMF keeps running fastest for on all the three datasets. For better comparison, the stopping condition of the three algorithms was set identically. Here we could discover that the convergence rate of GSNMF is must faster than NeNMF and consistently faster than MUR, and the computing consumption of GSNMF is consistently the least.

4. CONCLUSION

This paper proposes a Gauss-Seidel based non-negative matrix factorization (GSNMF) method. GSNMF iteratively projects the samples onto the subspaces spanned by each factor matrix and makes the updates another factor matrix by solving a linear system with the Gauss-Seidel method. By analyzing the error bound between GSNMF and NMF, we show that GSNMF can obtain a local minima of NMF. GSNMF can reduce the influence of imbalance between number of genes and number of samples, and thus outperforms NMF for gene expression clustering.

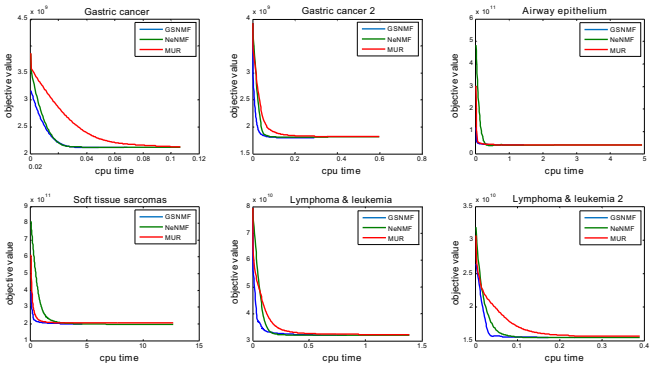

Fig. 3. The average time consumption plots for the three NMF algorithms over 100 runs on all six gene expression datasets: gastric cancer, gastric cancer 2, airway epithelium database, soft tissue sarcomas, lymphoma & leukemia, lymphoma & leukemia 2.

5. ACKNOWLEDGEMENT

This work was supported by The National Natural Science Foundation of China (under grant No. U1435222 and No. 61502515). And this work was also supported in part by grants from 973 project 2013CB329006, RGC under the contract CERG 16212714.

6. REFERENCES

[1] Patrick O Brown and David Botstein, "Exploring the new world of the genome with dna microarrays," *Nature genetics*, vol. 21, pp. 33–37, 1999.

[2] Anbupalam Thalamuthu, Indranil Mukhopadhyay, Xiaojing Zheng, and George C Tseng, "Evaluation and comparison of gene clustering methods in microarray analysis," *Bioinformatics*, vol. 22, no. 19, pp. 2405–2412, 2006.

[3] Stephen C Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

[4] Michael B Eisen, Paul T Spellman, Patrick O Brown, and David Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proceedings of the National Academy of Sciences*, vol. 95, no. 25, pp. 14863–14868, 1998.

[5] James MacQueen et al., "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Oakland, CA, USA., 1967, vol. 1, pp. 281–297.

[6] John A Hartigan and Manchek A Wong, "Algorithm as 136: A k-means clustering algorithm," *Applied statistics*, pp. 100–108, 1979.

[7] Teuvo Kohonen, "Self-organized formation of topologically correct feature maps," *Biological cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[8] Teuvo Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1, pp. 1–6, 1998.

[9] Pablo Tamayo, Donna Slonim, Jill Mesirov, Qing Zhu, Sutisak Kitareewan, Ethan Dmitrovsky, Eric S Lander, and Todd R Golub, "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation," *Proceedings of the National Academy of Sciences*, vol. 96, no. 6, pp. 2907–2912, 1999.

[10] Karthik Devarajan, "Nonnegative matrix factorization: an analytical and interpretive tool in computational biology," *PLoS Computational Biology*, vol. 4, no. 7, pp. e1000029, 2008.

[11] Jean-Philippe Brunet, Pablo Tamayo, Todd R Golub, and Jill P Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," *Proceedings of the national academy of sciences*, vol. 101, no. 12, pp. 4164–4169, 2004.

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

[13] Chun-Hou Zheng, De-Shuang Huang, Lei Zhang, and Xiang-Zhen Kong, "Tumor clustering using nonnegative matrix factorization with gene selection," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 4, pp. 599–607, 2009.

[14] Daniel D Lee and H Sebastian Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[15] Daniel D Lee and H Sebastian Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[16] JM Bernardo, MJ Bayarri, JO Berger, AP Dawid, D Heckerman, AFM Smith, and M West, "Bayesian factor regression models in the large p, small n?paradigm," *Bayesian statistics*, vol. 7, pp. 733–742, 2003.

[17] Richard Barrett, Michael W Berry, Tony F Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk Van der Vorst, *Templates for the solution of linear systems: building blocks for iterative methods*, vol. 43, Siam, 1994.

[18] William Morton Kahan, *Gauss-Seidel methods of solving large systems of linear equations*, Ph.D. thesis, Thesis–University of Toronto, 1958.

[19] Chuan-bi Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[20] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan, "Manifold regularized discriminative non-negative matrix factorization with fast gradient descent," *Image Processing, IEEE Transactions on*, vol. 20, no. 7, pp. 2030–C2048, 2011.

[21] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan, "Nenmf: an optimal gradient method for nonnegative matrix factorization," *Signal Processing, IEEE Transactions on*, vol. 60, no. 6, pp. 2882–2898, 2012.