

Multiplicative Algorithms for Constrained Non-negative Matrix Factorization

Chengbin Peng*, Ka-Chun Wong[†], Alyn Rockwood*, Xiangliang Zhang*, Jinling Jiang[‡], David Keyes*

*Computer, Electrical and Mathematical Sciences & Engineering,

King Abdullah University of Science and Technology, Thuwal, Kingdom of Saudi Arabia,

{chengbin.peng, alyn.rockwood, xiangliang.zhang, david.keyes}@kaust.edu.sa

[†]Department of Computer Science, University of Toronto, Toronto, Canada, wkc@cs.toronto.edu

[‡]Department of Computer Science, Aalborg University, Aalborg, Denmark, jinling@cs.aau.dk

Abstract—Non-negative matrix factorization (NMF) provides the advantage of parts-based data representation through additive only combinations. It has been widely adopted in areas like item recommending, text mining, data clustering, speech denoising, etc. In this paper, we provide an algorithm that allows the factorization to have linear or approximately linear constraints with respect to each factor. We prove that if the constraint function is linear, algorithms within our multiplicative framework will converge. This theory supports a large variety of equality and inequality constraints, and can facilitate application of NMF to a much larger domain. Taking the recommender system as an example, we demonstrate how a specialized weighted and constrained NMF algorithm can be developed to fit exactly for the problem, and the tests justify that our constraints improve the performance for both weighted and unweighted NMF algorithms under several different metrics. In particular, on the Movielens data with 94% of items, the Constrained NMF improves recall rate 3% compared to SVD50 and 45% compared to SVD150, which were reported as the best two in the top-N metric.

Keywords—Non-negative Matrix Factorization; Linear Constraints; Multiplicative Algorithm;

I. INTRODUCTION

Non-negative matrix factorization (NMF) [1], [2] has been proposed for more than a decade. Unlike traditional matrix factorization methods, in this algorithm no entries of the matrices are less than 0. NMF decomposes a non-negative matrix approximately into the product of two lower-ranked non-negative matrices. One advantage is that it enables parts-based representation by additive only combinations [3]. NMF has better clustering capabilities as well compared to traditional K-means [4]. A typical approach to NMF is iterative: alternatively fix one of the two factors and calculate the other in successive iterations [1], [2]. The simplicity and efficiency of this approach has made it widely adopted in areas such as face recognition [3], text mining [3], [5], data clustering [4], graph mining [6], [7], privacy protection [8], speech denoising [9], item recommending [10], [11], etc.

Specifically, Ref. [12] proposed a label based constraint on the objective function of the factorization algorithm. Ref. [4] introduced orthogonal nonnegative matrix tri-factorization, in which the tightness of the constraint is fixed by the algorithm. However, different application domains usually have different requirements for the NMF algorithm. Therefore, an

algorithm allowing problem-oriented constraints is eagerly sought. In this research, we introduce a framework that can accept a variety of equality and inequality constraints in the NMF algorithms, and the tightness of these constraints is adjustable. We prove that if the constraint functions are linear, algorithms following our the multiplicative framework must converge. This property can greatly facilitate the adoption of NMF algorithm in many application fields requiring different variations of constraints.

As an example within the framework, we developed two special constraints in Weighted and Constrained Non-negative Matrix Factorization (WC-NMF) based on W-NMF [11] and the Constrained Non-negative Matrix Factorization (C-NMF) based on NMF [1]. These two constraints enhance the formula by maintaining the upper bound of the missing entries, as well as approximately orthogonalizing factor matrices.

We use several metrics for comparison in the recommender system example. Our C-NMF and WC-NMF algorithms significantly and stably improve the performance of NMF algorithms in top-N metric, and slightly improve them in terms of Root-Mean-Square Error (RMSE) [13] and Normalized Mean Absolute Error (NMAE) [11], [14] using a squeezed lower-bound strategy. On the Movielens data without the 6% most popular items, C-NMF achieves a much better result in top-N metric than SVD50 and SVD150, the most predictive method in Ref. [15].

The remainder of the paper is organized as follows: Section II describes the algorithm of WC-NMF, and provides the convergence guarantee for a group of NMF problem with a variety of linear constraints. The implementation example and the performance comparison are in Section III and Section IV respectively. Section V summarizes and looks ahead for C-NMF and WC-NMF. Notations are defined in Table I.

II. WEIGHTED AND CONSTRAINED NON-NEGATIVE MATRIX FACTORIZATION (WC-NMF)

A. Problem Definition

A typical dimension reduction problem through non-negative matrix factorization (NMF) can be written as $V \approx WH$, where entries in W and H are all non-negative.

Table I
NOTATIONS

Notation	Definition	Example & Remark
$\{\cdot\}_{i,j}$	The value of entry (i, j) of the anterior matrix.	Similarly, $\{\cdot\}_i$ is the i th element for a vector.
\times	Element-wise multiplication.	For vectors and matrices.
\cdot	Vinculum, element-wise division.	For vectors and matrices.
$[\cdot]^+$	The positive part of the anterior matrix.	Given a real matrix A , A^+ is the matrix whose entries are $A^+_{i,j} = \begin{cases} A_{i,j}, & \text{if } A_{i,j} > 0 \\ 0, & \text{otherwise} \end{cases}$. If h is a vector, $\nabla F(h)^+$ and $\nabla^2 F(h)^+$ are the positive parts of the gradient vector and the hessian matrix respectively.
$[\cdot]^-$	The negative part of the anterior matrix.	Given a real matrix A , A^- is the matrix whose entries are $A^-_{i,j} = \begin{cases} -A_{i,j}, & \text{if } A_{i,j} < 0 \\ 0, & \text{otherwise} \end{cases}$ so that $A = A^+ - A^-$. Similarly, $\nabla F(h)^-$ and $\nabla^2 F(h)^-$ are the corresponding negative parts.
$[\cdot]^{(i)}$	Different values of i indicate different variables or functions.	$A^{(1)}$ and $A^{(2)}$ could be different matrices, but of the same meaning.
$\ \cdot\ $	Matrix norm.	For simplicity, we use Frobenius norm in this paper.
$\nabla_x F(\cdot)$	The derivative of function F with respect to matrix x .	If $F(x) = \ Ax + B\ ^2$, $\nabla_x F(x) = A^T(Ax + B)$. If $F(x) = \ xA + B\ ^2$, $\nabla_x F(x) = (xA + B)A^T$.

Often, the matrix V contains some unknown entries, and a binary matrix M can be adopted whose entry equals 0 only if the corresponding entry of V is unknown. Therefore, the original factorization becomes $M \times V \approx M \times (WH)$, which can be solved by Weighted Non-negative Matrix Factorization (W-NMF) [11]. We note that NMF is a special case of W-NMF in which all the entries of M are 1.

Assume the data matrix is $V \in \mathbb{R}^{m \times n}$, and we want to represent it by the product of two lower dimensional matrices $W \in \mathbb{R}^{m \times k}$ and $H \in \mathbb{R}^{k \times n}$.

The NMF problem with weighting is

$$\min_{W \geq 0, H \geq 0} \|M \times [V - (WH)]\| \quad (1)$$

in which $M \in \mathbb{R}^{m \times n}$, and \geq is an element-wise operator.

Ref. [11] provides the iterative formula, and we demonstrate herein that as a special case of our algorithm, it is a method with convergence guaranteed.

B. Algorithm

Let $J(W, H) = \|M \times [V - (WH)]\|^2$. Suppose we have a multiple constrained NMF algorithm as follows.

Algorithm

Input: Training set V , Integer k , Constraint function C

Output: Factorization result W and H

1. initialize W and H ;
2. **repeat**
- 3.

$$W \leftarrow \arg \min_{W \geq 0} J(W, H) \quad (2)$$

s.t. $C^{(i)}(W, H) = 0, i = 1, 2, 3, \dots$

$$H \leftarrow \arg \min_{H \geq 0} J(W, H) \quad (3)$$

s.t. $C^{(i)}(W, H) = 0, i = 1, 2, 3, \dots$

4. **until** converged

By modifying the constraint function slightly, the Lagrange can be written as $F(W, H) = J(W, H) + \sum_i \lambda_i \|C^{(i)}\|^2$. Hence, we have the following update rules in each iteration:

$$W \leftarrow W \times \frac{\nabla_W J(W, H)^- + \sum_i \lambda_i [\nabla_W \|C^{(i)}\|^2]^-}{\nabla_W J(W, H)^+ + \sum_i \lambda_i [\nabla_W \|C^{(i)}\|^2]^+} \quad (4)$$

$$H \leftarrow H \times \frac{\nabla_H J(W, H)^- + \sum_i \lambda_i [\nabla_H \|C^{(i)}\|^2]^-}{\nabla_H J(W, H)^+ + \sum_i \lambda_i [\nabla_H \|C^{(i)}\|^2]^+} \quad (5)$$

If there are inequalities in the algorithm, we can use a non-negative slackness variable S to transform the inequality to be equality. For example, given the constraint $C^{(j)} \leq 0$, the corresponding S should satisfy $C^{(j)} + S^{(j)} = 0$, so $C^{(j)}$ in Eq. (4) and (5) should be replaced by $C^{(j)} + S^{(j)}$ with an additional update rule in each iteration:

$$S^{(j)} \leftarrow S^{(j)} \times \frac{[C^{(j)}]^-}{[C^{(j)}]^+ + S^{(j)}} \quad (6)$$

In practice, we should also add a very small epsilon to the numerator and the denominator in the update rules to maintain the numerical stability.

C. Convergence Analysis of the Multiplicative Algorithms

In this section, we provide a convergence proof for the above multiplicative algorithm. In this algorithm, the objective functions F can be written as a quadratic function. Without loss of generality, let the argument h be a column vector of length k . If parameter matrix of the quadratic term is non-negative, we have

$$F(h) = h^T Y^+ h + Z^+ h - Z^- h \quad (7)$$

where Y^+ , Z^+ and Z^- are all non-negative matrices, and especially, as Y is a covariance matrix, Y^+ is symmetric. Then we can divide the gradient of F into two parts:

$$\nabla F(h) = \nabla F(h)^+ - \nabla F(h)^- \quad (8)$$

where $\nabla F(h)^+ = 2Y^+ h + Z^+$ and $\nabla F(h)^- = Z^-$.

Now we need to prove the following theorem.

Theorem 1: The objective function $F(h)$ in Eq. (7) is non-increasing if its argument h is updated according to the following rule:

$$h_t \leftarrow h_{t-1} \times \frac{\nabla F(h_{t-1})^-}{\nabla F(h_{t-1})^+} \quad (9)$$

Proof:

In the proof, we use $\text{diag}(\cdot)$ to represent a square matrix whose main diagonal is the vector inside the parentheses. Assume v is an arbitrary vector of length k . Then we have

$$\begin{aligned} & v^T [\text{diag}(\frac{\nabla F(h)^+}{h}) - \nabla^2 F(h)^+] v \\ &= \sum_{i=1}^n (v_i^2 \frac{\sum_{j=1}^n 2Y_{i,j}^+ h_j + Z_i^+}{\{h\}_i}) - \sum_{i=1}^n \sum_{j=1}^n 2(v_i Y_{i,j}^+ v_j) \\ &\geq 2 \sum_{i=1}^n (v_i^2 \frac{\sum_{j=1}^n Y_{i,j}^+ h_j}{\{h\}_i}) - 2 \sum_{i=1}^n \sum_{j=1}^n (v_i Y_{i,j}^+ v_j) \\ &\geq 0 \end{aligned} \quad (10)$$

where the last step can be proved by multiplying $\text{diag}(h)$ on both side of the formula (without changing the positive definiteness), and obtain a quadratic expression.

Consequently, as $\nabla^2 F(h)^-$ is always a zero matrix according to the definition in Eq. (7), we have

$$\begin{aligned} & v^T [\text{diag}(\frac{\nabla F(h)^+}{h}) - \nabla^2 F(h)] v \\ &= v^T [\text{diag}(\frac{\nabla F(h)^+}{h}) - \nabla^2 F(h)^+] v + v^T [\nabla^2 F(h)^-] v \\ &\geq 0 \end{aligned} \quad (11)$$

which insures that the matrix $\text{diag}(\frac{\nabla F(h)^+}{h}) - \nabla^2 F(h)$ is positive semi-definite.

Then we can use the idea of an auxiliary function [1] for further results. First we define a function

$$\begin{aligned} G(h^{(1)}, h^{(2)}) &= F(h^{(2)}) + (h^{(1)} - h^{(2)})^T \nabla F(h^{(2)}) \\ &\quad + 0.5(h^{(1)} - h^{(2)})^T \text{diag}(\frac{\nabla F(h^{(2)})^+}{h^{(2)}})(h^{(1)} - h^{(2)}) \end{aligned} \quad (12)$$

By comparing it with Taylor series of the objective function

$$\begin{aligned} F(h^{(1)}) &= F(h^{(2)}) + (h^{(1)} - h^{(2)})^T \nabla F(h^{(2)}) \\ &\quad + 0.5(h^{(1)} - h^{(2)})^T \nabla^2 F(h^{(2)})(h^{(1)} - h^{(2)}) \end{aligned} \quad (13)$$

and according to the result in Eq. (11), we find it always holds that

$$G(h^{(1)}, h^{(2)}) \geq F(h^{(1)}) \quad (14)$$

It is also easy to see that

$$G(h^{(2)}, h^{(2)}) = F(h^{(2)}) \quad (15)$$

Therefore, $G(h^{(1)}, h^{(2)})$ is an auxiliary function for $F(h^{(1)})$. Hence, $F(h)$ is non-increasing by taking the following update rule

$$h_t \leftarrow \arg \min_h G(h, h_{t-1}) \quad (16)$$

according to the results that $F(h_t) \leq G(h_t, h_{t-1})$ by Eq. (14), $G(h_{t-1}, h_{t-1}) = F(h_{t-1})$ by Eq. (15) and $G(h_t, h_{t-1}) \leq G(h_{t-1}, h_{t-1})$ by Eq. (16), similar as Lemma 1 in Ref. [1].

Finally, according to the definition of G , we have

$$\begin{aligned} h_t &\leftarrow h_{t-1} - [\text{diag}(\frac{\nabla F(h_{t-1})^+}{h_{t-1}})]^{-1} \nabla F(h_{t-1}) \\ &= h_{t-1} \times \frac{\nabla F(h_{t-1})^-}{\nabla F(h_{t-1})^+} \end{aligned} \quad (17)$$

Iterative methods described in Eq. (23), (26), and (28) are in the framework of our updating rule, but in element format

$$\{h_t\}_a \leftarrow \{h_{t-1}\}_a \times \frac{\{\nabla F(h_{t-1})^-\}_a}{\{\nabla F(h_{t-1})^+\}_a} \quad (18)$$

If the parameter matrix Y of the quadratic term has negative entries, a non-negative slackness may be added to both positive and negative parts of the matrix so the that update rule can still work according to Eq. (11).

When the argument h is a row vector, the proof is almost the same by simply considering h^T as an argument for function F . The vector version can also be extended easily to a matrix version.

III. SAMPLE IMPLEMENTATIONS

In this section, we give two sample constraints for the weighted and constrained NMF problem. The first is a matrix of upper bounds U for entries in V . The second is to require H to be an orthogonal matrix: $HH^T = I$.

These two constraints can be applied independently, by choosing appropriate parameters. The first constraint should be derived from the property of the problem itself, while the second is established for the ease of analyzing the basis components.

Algorithm WC-NMF

(* Weighted and Constrained Non-negative Matrix *)
(* Factorization *)

Input: Training set V , Integer k , Upper bound U

Output: Factorization result W and H

1. initialize W and H ;
2. **repeat**
- 3.

$$\begin{aligned} W &\leftarrow \arg \min_{W \geq 0} \|M \times [V - (WH)]\|^2 \\ &\text{s.t. } WH \leq U \end{aligned} \quad (19)$$

$$\begin{aligned} H &\leftarrow \arg \min_{H \geq 0} \|M \times [V - (WH)]\|^2 \\ &\text{s.t. } HH^T = I \end{aligned} \quad (20)$$

4. **until** converged

In contrast to Eq. (20), in Ref. [4] the orthogonalization part constrains only the trace, while in our algorithm, the orthogonalization takes the whole matrix into consideration.

Theoretically, compared to NMF algorithms without constraints [1], [11], the computing time is almost doubled. If only taking the orthogonalization into account, the time complexity is similar to the algorithm in Ref. [4].

A. Procedure to Calculate H in Eq. (20)

By rewriting the optimization problem in Lagrangian function, we get the following expression with a Lagrange multiplier [4] equal to $0.5\lambda_H$:

$$H = \arg \min_{H \geq 0} \|M \times [V - (WH)]\|^2 + 0.5\lambda_H \|HH^T - I\|^2 \quad (21)$$

We approximate the quadratic term in the constraint by a linear expression $HH_{(t-1)}^T - I$ where the notation $H_{(t-1)}$ represents the value of the unknown variable H in the previous iteration. This approximation can be interpreted as an expectation maximization method [11].

The derivative of the second term is

$$\begin{aligned} & \frac{d(0.5\lambda_H \|HH_{(t-1)}^T - I\|^2)}{dH} \\ &= \lambda_H HH_{(t-1)}^T H_{(t-1)} - \lambda_H H_{(t-1)} \end{aligned} \quad (22)$$

For simplicity, by omitting the subscript $(t-1)$ for $H_{(t-1)}$ and following the approach of Eq. (5), we get

$$\{H\}_{a,i} \leftarrow \{H\}_{a,i} \frac{\{W^T(M \times V) + \lambda_H H\}_{a,i}}{\{W^T[M \times (WH)] + \lambda_H HH^T\}_{a,i}} \quad (23)$$

B. Procedure to Calculate W in Eq. (19)

First, we use a slackness variable S for the inequality in the constraints of Eq. (19), so we have a new but equivalent expression:

$$W = \arg \min_{W \geq 0, S \geq 0} \|M \times [V - (WH)]\|^2, \text{ s.t. } WH + S = U \quad (24)$$

In terms of the Lagrange function, it is

$$\begin{aligned} W = \arg \min_{W \geq 0, S \geq 0} & \|M \times V - M \times (WH)\|^2 \\ & + 0.5\lambda_W \|WH + S - U\|^2 \end{aligned} \quad (25)$$

Similarly, we have a multiplicative expression for W

$$\{W\}_{i,a} \leftarrow \{W\}_{i,a} \frac{\{(M \times V)H^T + \lambda_W UH^T\}_{i,a}}{\{M \times (WH)H^T + \lambda_W (WH + S)H^T\}_{i,a}} \quad (26)$$

As we want to minimize the error term due to the slackness, we have

$$S = \arg \min_{S \geq 0} \|WH + S - U\| \quad (27)$$

Table II
LIST OF ALGORITHMS FOR COMPARISON

Without Weights	
Algorithm Name	Description
C-NMF	Our algorithm, NMF only with constraints but no weights.
Pure-NMF	Pure NMF algorithm [1].
EM-NMF	Expectation maximization with NMF [11]. In the test, the numbers of outer and inner iterations are both 10, and pure-NMF [1] is used for inner iteration.
Orth-NMF	NMF with orthogonalization, initialized by K-means [4].
Pure-SVD	Recommender simply by SVD decomposition [15].
With Weights	
Algorithm Name	Description
WC-NMF	Our algorithm, NMF with weights and constraints.
W-NMF	Weighted NMF [11].
Hybrid NMF	run EM NMF for initialization, and then run W-NMF [11].

Hence,

$$\{S\}_{i,a} \leftarrow \{S\}_{i,a} \frac{\{U\}_{i,a}}{\{WH + S\}_{i,a}} \quad (28)$$

Usually, W and H are randomly initialized, but when S is involved and initialized by $S = U - WH$, W need to be scaled to make sure that $U \geq WH$.

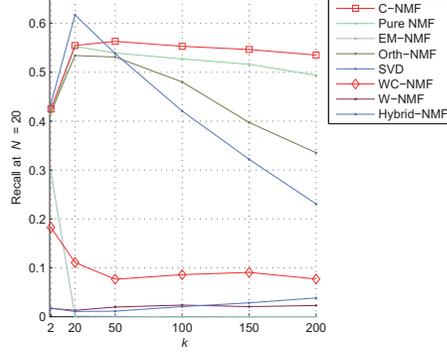
IV. APPLICATION ON MOVIELENS RATING DATA

MovieLens data was adopted by a number of researchers as a benchmark for their work [10], [11], [15], [16]. In this work, we used the data set called MovieLens 100k¹, which consists of 100,000 ratings from 1000 users on 1700 movies. The smallest rating value in the data set is 1, and the largest is 5. During the test, these data were divided into two sets: about 98.6% for training, and about 1.4% for testing set.

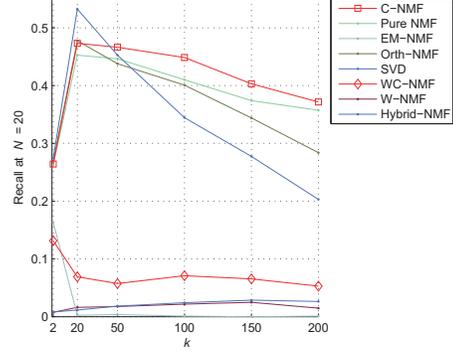
We listed the algorithms for comparison in Table II. There are also some implementation details. The number of iterations are all 30, if not explicitly indicated. In C-NMF, we simply use Algorithm *WC-NMF* by setting all the entries in W to be 1, with $U = 5$, $\lambda_W = 1e - 4$ and $\lambda_H = 1$. In WC-NMF, we use the full version of Algorithm *WC-NMF* where W serves as a mask, with $U = 5$, $\lambda_W = 1e - 4$ and $\lambda_H = 0.1$. Each result is averaged from running for 10 times. In each run, the initial values for different algorithms are the same.

There are several metrics for measuring recommender system algorithms. Among those, the top-N method measures the probability that a 5-score testing movie can be in the list when obtaining the N top-ranked movies according to the algorithm [15]. This measurement has its practical significance, because a recommender system is usually asked to provide a few items that appeal to the user. The traditional

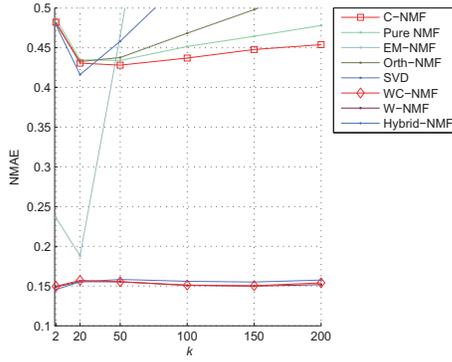
¹<http://www.grouplens.org/node/73>



(a) top-N, at $N = 20$

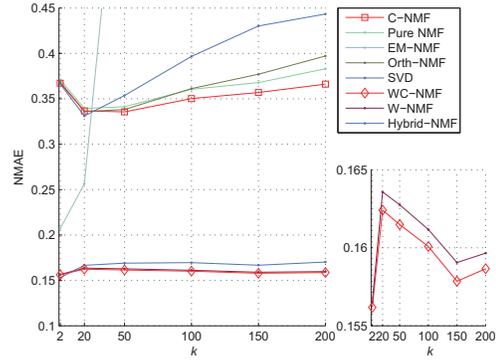


(a) top-N, at $N = 20$



(b) NMAE versus k

Figure 1. Tests on Data



(b) NMAE versus k when Lower Bound is Squeezed

Figure 2. Tests on Data with Most Popular Items Removed

element-wise comparisons like RMSE and NMAE are made as well.

Generally, the algorithms without weights perform better in top-N metric, while those with weights perform better in RMSE and NMAE.

In top-N when $k \geq 50$, our algorithm C-NMF performs better than all the other methods, including pure SVD, the best algorithm at $k = 50$ in Ref. [15]; when $k \geq 20$, our algorithm WC-NMF performs much better than other weighted algorithms. In NMAE (Fig. 1b) and RMSE, among the algorithms without weights, when $k \geq 50$, C-NMF is the best, while the performances of the weighted algorithms are similar.

Ref. [15] claims that some popular items may mislead the top-N result. In their Movielens data set, about 6% of the most popular movies containing 33% of the total ratings, and when these movies are removed, SVD performs the best when $k = 50$ and $k = 150$ respectively. Therefore, we launch another test where we remove the hottest movies containing 33% of the total ratings, and used the same parameters as in the first test. From Fig. 2(a) and 3, we can see that in top-N, C-NMF still performs better than all the

others including SVD when $k \geq 50$. Especially compared to SVD in Fig. 3, it outperforms 3% at $k = 50$ and 45% at $k = 150$ when $N = 20$. Also in top-N, WC-NMF is the best within the weighted category.

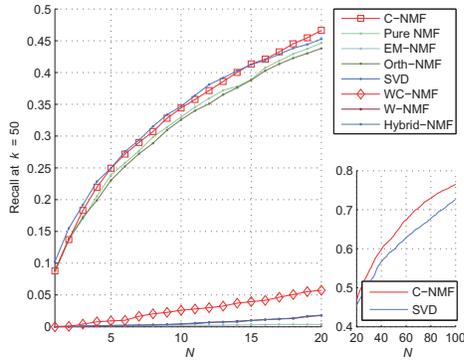
To fully utilize the advantage of our upper-bound constraint, we have also tried to squeeze the lower bound of the data to be 0. With appropriate parameter setting, it achieves a similar result.

In summary, if top-N metric is emphasized, C-NMF is an ideal choice, and if all the three metrics are important, WC-NMF is good especially when the lower bound of the data is squeezed.

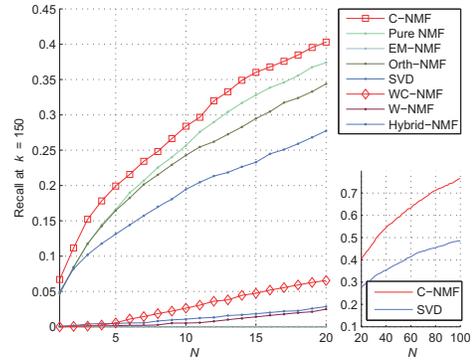
V. DISCUSSION AND CONCLUSION

In this research, we devise a framework for both Weighted and Constrained Non-negative Matrix Factorization and Constrained Non-negative Matrix Factorization. Our contributions are as follows:

We provide a convergence guaranteed multiplicative method for NMF with general formulation of constraints, only if the constraint function is linear with respect to each of the factors. Our approach also allows the freedom to choose a suitable tightness for each constraint when



(a) Recall versus N at $k = 50$



(b) Recall versus N at $k = 150$

Figure 3. Top- N test on Data with Most Popular Items Removed

embedded into the objective function. These two properties can facilitate the application of NMF method in different domains.

In addition, we give examples of specialized constraints. The first constraint is to limit the upper bounds for unknown entries during the factorization, and the second is to orthogonalize the factor matrices. Taking the recommender system as an example, our algorithm can achieve excellent results in top- N metric, and can gain small improvements under other metrics.

In the future, we will focus on using different optimization methods to achieve a better convergence rate for this kind of NMF methods.

REFERENCES

- [1] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, 2001.
- [2] C. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural computation*, vol. 19, no. 10, pp. 2756–2779, 2007.
- [3] D. Lee, H. Seung *et al.*, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.
- [4] C. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix tri-factorizations for clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 126–135.
- [5] W. Xu, X. Liu, and 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.
- [6] Y. Mao and L. Saul, "Modeling distances in large-scale networks by matrix factorization," in *Proceedings of the 4th ACM SIGCOMM conference on Internet Measurement*. ACM, 2004, pp. 278–287.
- [7] C. Ding, X. He, and H. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. SIAM Data Mining Conf*, no. 4, 2005, pp. 606–610.
- [8] J. Wang, W. Zhong, and J. Zhang, "Nnmf-based factorization techniques for high-accuracy privacy protection on non-negative-valued datasets," in *Data Mining Workshops, 2006. ICDM Workshops 2006. Sixth IEEE International Conference on*. IEEE, 2006, pp. 513–517.
- [9] M. Schmidt, J. Larsen, and F. Hsiao, "Wind noise reduction using non-negative sparse coding," in *Machine Learning for Signal Processing, 2007 IEEE Workshop on*. IEEE, 2007, pp. 431–436.
- [10] G. Chen, F. Wang, and C. Zhang, "Collaborative filtering using orthogonal nonnegative matrix tri-factorization," *Information Processing & Management*, vol. 45, no. 3, pp. 368–379, 2009.
- [11] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," *SIAM*, 2006.
- [12] H. Liu, Z. Wu, D. Cai, and T. Huang, "Constrained non-negative matrix factorization for image representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 99, pp. 1–1, 2011.
- [13] G. Takács, I. Pilászy, B. Németh, and D. Tikk, "Matrix factorization and neighbor based algorithms for the netflix prize problem," in *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008, pp. 267–274.
- [14] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Information Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.
- [15] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 39–46.
- [16] W. Zeng, Y. Zhu, L. Lü, and T. Zhou, "Negative ratings play a positive role in information filtering," *Physica A: Statistical Mechanics and its Applications*, 2011.