Simplicial Nonnegative Matrix Factorization

Duy Khuong Nguyen, Khoat Than, Tu Bao Ho
Japan Advanced Institute of Science and Technology
Email: {khuongnd, khoat, bao}@jaist.ac.jp

***Abstract*—Nonnegative matrix factorization (NMF) plays a crucial role in machine learning and data mining, especially for dimension reduction and component analysis. It is employed widely in different fields such as information retrieval, image processing, etc. After a decade of fast development, severe limitations still remained in NMFs methods including high complexity in instance inference, hard to control sparsity or to interpret the role of latent components. To deal with these limitations, this paper proposes a new formulation by adding simplicial constraints for NMF. Experimental results in comparison to other state-of-the-art approaches are highly competitive.**

I. Introduction

Many algorithms in data mining cannot deal with large datasets because of their rawness, high dimension and complex distribution. To deal with this situation, two fundamental purposes have been raised in data processing: Transforming the data into a lower dimension space and extracting latent components and variables inside the datasets to represent data [1]. Nonnegative matrix factorization (NMF) is one of the most popular effective methods to pursue these two purposes. Many datasets in various fields have been formed as matrices with nonnegative values. NMF aims to factorize such a matrix X into a product of two matrices, $X \approx FG$, where G contains basic vectors in the new space and F contains new corresponding coefficients of data instances in X. In other words, this factorization transforms data instances into new space of basic vectors.

Many NMF methods have been developed in the last decade by using divergence functions, constraints and regularizations. Initially, basic NMF [2] only allows approximating the original nonnegative data by a product of two matrices. By this approximation, each object is represented as a non-subtractive combinations of nonnegative parts or basic vectors. Following [2] many algorithms were proposed for different divergence functions [3]. Currently, by adapting requirements for data analysis problems and data types, many variants of NMF are being developed. Also, various new divergence functions are employed [4], and local constraints are added to improve the quality of matrix decompositions, which preserve the local features [5]. Sparse representation can be achieved by adding sparseness constraints [6], [7]. In addition, some work has implicitly or explicitly added orthogonality constraints [8], [9].

Usually, NMF is solved by iterative multiplicative updating algorithms. However, minimizing object functions does not guarantee a unique solution. The existence of many local minima solutions makes the algorithms suffering from rotational ambiguities. If no prior information is known, the normalization of rows in G or latent components will help to reduce the effects of these ambiguities [10]. Particularly, if $X \approx FG$ is a solution, $X \approx F[D_G G'] = [FD_G]G'$ is also a solution; where $D_G = diag(||G_{1\bullet}||_p^{-1}, ..., G_{K\bullet}||_p^{-1}), \quad p \in [0, \infty)$ and $G = D_G G'$. As a result, the latent components are normalized as basic vectors. Although this technique is advantageous to optimize the objective functions, the role of basic vectors is not easy to interpret directly via their coefficients.

In this work, we propose a new NMF formulation by adding new prior information into NMF, in which each data instance is a convex combination of the latent components. In other words, each instance is a probabilistic distribution over the latent components. By this way, we associate a probabilistic model with the NMF problem. As a result, we obtain more advantages than the previous formulations such as easy interpretability, low complexity, convexity, sparsity, and distributability and parallelability. We also develop effective algorithms for the two most popular divergence functions: squared Euclidean distance and *KL*-divergence [4].

II. New Formulation

Mathematically, we can define the NMF problem as follows:

Definition 1 [NMF]: *Given a dataset consisting of M N-dimension vectors $X = [X_1, X_2, ..., X_M] \in R_+^{M \times N}$, where each vector presents a data instance. NMF seeks to decompose X into a product of two nonnegative factorizing matrices F and G, where $F = [F_1, ..., F_M] \in R_+^{M \times K}$ and $G = [G_1, ..., G_K] \in R_+^{K \times N}$ are coefficient matrix and latent component matrix, respectively, $X \approx FG$.*

We assume that each instance is a convex combination of the latent components obtained by adding a new simplicial constraint into NMF. Hence, we have:

Definition 2 [Simplicial NMF]: *Simplicial NMF is NMF where each instance X_m is a convex combination of the latent components $X_m \approx \sum_{k=1}^{K} F_{mk} G_k$ and $\sum_{k=1}^{K} F_{mk} = 1$ for all m.*

By adding this new constraint, we have associated a probabilistic model with NMF problem, in which each instance is a probabilistic distribution over the latent components and represented as a convex combination of latent components. In other words, this convex combination provides explicitly the extent of contribution of each latent component, while other formulations of NMF do not have. Moreover, regarding to geometry meaning, each instance is projected as a point on the simplex of latent components. This projection is called instance inference. As a result, we obtained significant properties:

- **Sparsity**: Instance inference is casted as a convex problem over the simplex of latent components by adding the simplicial condition. Furthermore, we can

easily control the solution sparsity via greedy approximation algorithms such as Frank-Wolfe algorithm [11].

- **Convexity**: Obviously, inferring an instance is to find an approximation of the convex combination that is a convex optimization problem [11], [12], [13].
- **Computation**: The instance representation can be considered as a projection on the simplex of the latent components. Hence, the inference based on this projection can be much faster than other formulations because of the simplicial constraint added[11], [12]. In comparison to other formulations, this one has significant computing advantages in the inference of instances, while the learning step is the same with the previous basic formulations because they solve the same optimization problem.
- **Interpretability**: The new formulation gives a more comprehensible interpretation of the important role of coefficients. Particularly, each data instance is a convex combination of the latent components, in which the sum of coefficients always equals to 1 through NMF. Hence, the important role of the latent components on instances can be concisely represented via values of coefficients. Otherwise, for other formulations, evaluating the contribution of components is forceful because of the lack of constraints between coefficients. Alternatively, a post-processing can be employed to find out the role of the latent components. However, it is independent and inconsistent with learning NMF model.
- **Distributability and parallelizability**: NMF problem contains two sub-problems: inference and learning, see section IV. The learning problem is the same with other formulations and can be solved by distributed algorithms [14]. Meanwhile, the inference one of our formulation can be solved by a much faster algorithm comparing to the others', and it can be parallelized [12]. This favor is hard to be reached in other formulations.

The cost function is specially determined on the used divergence function. In this paper, we focus on solving this problem with the two most popular divergence functions with squared *Euclidean* distance and *KL*-divergence.

III. Divergence Functions and Problems

To control the quality of NMF, various cost functions are employed. The cost functions $f(X||FG)$ often contain two parts: The first part is a divergence function that measures the distance between original coordinates (X) and inverted coordinates (FG); and the second one is possibly regularizations and constraints to control sparsity or orthogonality.

A. Divergence Functions

Recently, there are numerous divergence functions, including squared *Euclidean* distance, *KL*-divergence, α-divergence, β-divergence, *IS* divergence, and *Bregman* divergence, etc. A chosen divergence mainly depends on the data type and its properties. The two most popular divergences are widely used in numerous applications:

- Squared *Euclidean* distance: $D(x||y) = ||x-y||_2^2 = \sum_i (x_i - y_i)^2$
- *KL*-divergence: $D(x||y) = \sum_i x_i.log\frac{x_i}{y_i} - x_i + y_i$, where x and y are positive vectors.

B. sNMF Problems

With these divergence functions, we have two basic problems of simplicial NMF (sNMF):

- sNMF with squared *Euclidean* distance
  $J(X||FG) = \sum_{m=1}^{M} D(X_m||F_mG)$
  where $D(X_m||F_mG) = ||X_m - F_mG||_2^2$
- sNMF with *KL*-divergence
  $J(X||FG) = \sum_{m=1}^{M} D(X_m||F_mG)$
  where $D(X_m||F_mG) = \sum_{n=1}^{N} (X_{mn}.log\frac{X_{mn}}{[F_mG]_n} - X_{mn} + [F_mG]_n)$; $X, F, G \geq 0$; $\sum_{k=1}^{K} F_{mk} = 1$ for all m.

IV. Algorithms

For solving sNMF, we employ iterative multiplicative updates like *EM* algorithm, which is presented in Algorithm 1. This algorithm contains two main steps: one for finding F when fixing G and the other for finding G when fixing F. In the first step, we find a set $F = \{f_m\}_{m=1}^{M}$, each of f_m is a probabilistic representation of data instances $\{x_m\}_{m=1}^{M}$. Hence, this step can be seen as *inference step* and the process is called as *inference*. In the other step called *learning step*, the latent components are acquired by minimizing $D(X||FG)$ when fixing F.

Algorithm 1 Nonnegative Matrix Factorization

Input: Data matrix $X = \{x_m\}_{m=1}^{M} \in R_+^{M \times N}$ and K.
Output: Coefficients $F = \{f_m\}_{m=1}^{M}$ and latent components $G = \{g_k\}_{k=1}^{K}$

```
Select randomly K components from M data instances
repeat
    Inference step: Fix G to find F by minimizing f(F) = J(X||FG) ;
    Learning step: Fix F to find G by minimizing f(G) = J(X||FG) ;
    if (convergence condition is satisfied) then
        break;
    end if
until False
```

A. Algorithms for sNMF with Squared Euclidean Distance

1. Inference Algorithm:

Remark 1. *The inference of data instances F in a new space of latent components by minimizing $J(X||FG)$ can be conducted independently.*

In this step, we need to minimize

$$J(X||FG) = \sum_{m=1}^{M} D(X_m||F_mG) = \sum_{m=1}^{M} ||X_m - F_mG||_2^2$$

Hence, since X and G are fixed in this step, minimizing $J(X||FG)$ is equivalent to minimizing $||X_m - F_mG||_2^2$ for each data instance m, which is performed independently by Algorithm 2.

Remark 2. *Inferring each data instance is equivalent to solving a convex optimization problem or a least square problem with simplicial condition.*

When set $x = X_m$ and $f = F_m$, the inference will become minimizing:

$$J(x||fG) = \sum_{i=1}^{N}(x_j - \sum_{k=1}^{K} f_k G_{kj})^2 \text{ where } \sum_{k=1}^{K} f_k = 1$$

This is a least square problem adding a simplicial constraint.

Moreover, adding the convex constraints leads to the existence of sparse solutions and it can avoid over-fitting problems. Specially, the convex constraints enable greedy algorithms, which is derived from Frank-Wolfe algorithm [11], in order to control directly and effectively sparsity of solutions, see more details in Algorithm 2.

Algorithm 2 Inference for data instance x

Input: Data instance x and latent components $G = \{g_k\}_{k=1}^K$
Output: New coefficient f minimizing $h = ||x - fG||_2^2$

Choose component g_k closest to x
Set $f = \mathbf{0}$; $f_k = 1$; and $r = x - g_k$
repeat
  Select $k = argmin_{k \in \{1..K\}}[\frac{\partial h}{\partial f}]_k$
  Select $\alpha = r(g_k - x)^T / ||g_k - x||_2^2$
  $\alpha = max(min(1, \alpha), -1)$
  $\alpha = max(\alpha, -(1-\alpha)f_k)$
  if $(\alpha == 0)$ **then**
    break;
  end if
  Set $r = x - \alpha g_k - (1-\alpha)(x - r)$
  Set $f = (1-\alpha)f$ and $f_k = f_k + \alpha$
until False

2. Learning Algorithm

Remark 3. *Learning components G by minimizing $J(X||FG)$ can be conducted independently in each column.*

We also have $J(X||FG) = \sum_{n=1}^{N} ||X_{\bullet n} - FG_{\bullet n}||_2^2$. Hence, minimizing $J(X||FG)$ is equivalent to minimizing $||X_{\bullet n} - FG_{\bullet n}||_2^2$ independently in each column n since X and F are fixed. Therefore, we can independently learn column coordinates of the latent components by solving nonnegative least-squares constraints problem [15].

B. Algorithms for sNMF with KL-divergence

1. Inference Algorithm

In the inference step, we need to find new coordinates $\{f_m\}_{m=1}^M$. Equivalent to the case for Euclidean distance, the inference in this divergence can be conducted independently for each data instance X_m. Moreover, inferring each data instance in KL-divergence is also solving a convex optimization problem with simplicial constraints and sparse properties like for *Euclidean* distance. In comparison to Algorithm 2, Algorithm 3 is more complicated because we cannot estimate directly the value α. Hence, α is seeked by binary search because $h(\alpha)$ is a continuous quadratic function.

Algorithm 3 Inference for data instance x

Input: Data instance x and latent components $G = \{g_k\}_{k=1}^K$
Output: New coefficient f minimizing

$$h(f) = \sum_{n=1}^{N}(x_n log \frac{x_n}{[fG]_n} - x_n + [fG]_n)$$

Choose component g_k closest to x
Set $f_i = \mathbf{0}$; $f_k = 1$
repeat
  Select $k = argmin_{i \in \{1..K\}}[\frac{\partial h}{\partial f}]_k$
  Select $\alpha = argmin_{\alpha \in [0,1]} h(\alpha g_k + (1-\alpha)fG)$
  Set $f = (1-\alpha)f$ and $f_k = f_k + \alpha$
until Convergence condition satisfied

2. Learning Algorithm

Equivalent to *Euclidean* distance, learning components can be conducted separately in each column because $J(X||FG) = \sum_{n=1}^{N} D(X_{\bullet n}||FG_{\bullet n})$, and X and F are fixed.

In this step, to approximate the solution $G_{\bullet n}$, we have $D(X_{\bullet n}||FG_{\bullet n}) = \sum_{m=1}^{M} X_{mn}\, log \frac{X_{mn}}{F_{m\bullet}G_{\bullet n}} - X_{mn} + F_{m\bullet}G_{\bullet n}$.

Hence, minimizing $D(X_{\bullet n}||FG_{\bullet n})$ is equivalent to minimizing

$$h(G_{1n}, ..., G_{kn}) = F_{m\bullet}G_{\bullet n} - \sum_{m=1}^{M} X_{mn}\, log F_{m\bullet}G_{\bullet n} \quad (1)$$

Moreover, based on Josen's inequality for the concave function log and non-negative coefficients $F_{m1}, ..., F_{mk}$ with $\sum_{k=1}^{K} F_{mk} = 1$, we have

$$log F_{m\bullet}G_{\bullet n} = log \sum_{K}^{k=1} F_{mk}G_{kn} \leq \sum_{k=1}^{K} F_{mk}\, log(G_{kn})$$

Then: $h(G_{1n}, ..., G_{kn}) \leq h'(G_{1n}, ..., G_{kn})$
$= \sum_{m=1}^{M}\sum_{k=1}^{K} F_{mk}g_{kn} - \sum_{m=1}^{M} X_{mn}\sum_{k=1}^{K} F_{mk} log(G_{kn})$.

$G_k n$ is approximated by minimizing $h'(G_{1n}, ..., G_{kn})$:

$$\frac{\partial h'}{\partial G_{kn}} = 0 \Leftrightarrow G_{kn} = \frac{\sum_{m=1}^{M} X_{mn}}{\sum_{m=1}^{M} F_{mk}} \quad (2)$$

Fortunately, in the learning step for KL-divergence, we can directly approximate the solution. Although this is only an approximate solution, it is really effective for KL-divergence and this technique has been employed in numerous applications.

V. Theoretical Analysis

As discussed above, the main difference between our algorithms and the others is in the inference step because we need to solve the same optimization problem in the learning step. Hence, we will only consider the complexity of inference, sparsity, and distributability and parallelizability.

A. Complexity

1. Complexity for sNMF with Squared Euclidean Distance

Theorem 1. *Consider Algorithm 2 to infer a data instance having N-dimension by K latent components with L iterations. Then its complexity is $O(L[K.S(N) + N])$, where $S(N)$ is a function estimate the number of non-zero elements in latent components and $S(N) \leq N$.*

Proof: In the Algorithm 2, for each iteration, we have:

$$\frac{\partial h}{\partial f} = 2(fG - X)G^T$$

Hence: $[\frac{\partial h}{\partial f}]_k = 2(fG - X)g_k^T = 2rg_k^T$

Therefore, the complexity of finding out the best coefficient k: $O(K.S(N))$. In addition, the complexity of estimating a is $O(N)$. Overall, the complexity for L iterations is $O(L[K.S(N) + N])$ ■

2. Complexity for sNMF with KL-divergence

Theorem 2. *Consider Algorithm 3 to infer a data instance having N-dimension by K latent components with L iterations. Then, its complexity is $O(L[K.S(N) + N. \log \frac{1}{\epsilon}])$.*

Proof: In the Algorithm 3, for each iteration, we have:

$$\frac{\partial h}{\partial f} = (\mathbf{1} - x./(fG))G^T$$

where $\mathbf{1} = (1, ..., 1)$ and $x./y = (\frac{x_1}{y_1}, ..., \frac{x_N}{y_N}) \in R^N$.

Hence: $[\frac{\partial h}{\partial f}]_k = (\mathbf{1} - x./fG)g_k^T$, and therefore, the complexity of finding out the best coefficient k is $O(K.S(N))$. In addition, the complexity of estimating $a \in [0, 1]$ is $O(N \log \frac{1}{\epsilon})$ where ϵ is a small positive quantity for the required precision, because we use binary search algorithm. Overall, the complexity for L iterations is $O(L[K.S(N) + N \log \frac{1}{\epsilon}])$ ■

The inference complexities are competitive with NMF using $Euclidean$ distance having $O(M.N^2)$ [13] and much better than other formulations with KL-divergence because we can not estimate directly solutions of inference and gradient methods does not work on this divergence [16].

In addition, for the learning step with KL-divergence, we employ an approximate algorithm with low complexity:

Theorem 3. *Let consider to learn new latent components after inferring coefficients of data instances. Then, its complexity is $O(M[S(N) + S(K)])$.*

Proof: This theorem is implied from formula 2 ■

2. Convergence Guarantee of Inference

Based on [11], we have

Theorem 4. *Let f be a twice differentiable convex function over simplex $\triangle$ and denote $C_f = sup_{y,z \in \triangle; \tilde{y} \in [y,z]}(y - z).\bigtriangledown^2 f(\tilde{y}).(y - z)^T$. After l iterations, the Frank-Wolfe algorithm will find an approximate solution x_l with at most $(l+1)$ non-zeros coefficients which satisfy*

$$max_{x \in \triangle} f(x_l) - f(x) \leq \frac{C_f}{l+1}$$

From this theorem, we have the following remarks:

- Convergence rate of inference is linear and the goodness of solutions is bounded, which are crucial in applications.
- Inference depends mostly on complexity of f and $\bigtriangledown f$.
- We can tradeoff easily between sparsity and quality of solutions by stop finding new latent components to optimize the cost function. This property is valid for real applications, which the number of non-zero coefficients is limited.

B. Sparsity

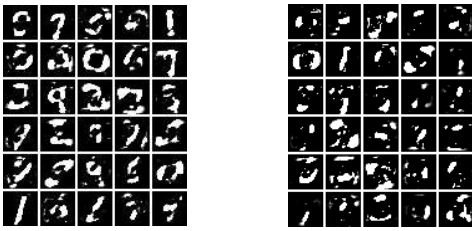
Recently, sparse solutions receive much interests in machine learning by its abilities of improving accuracy and saving storage with low complexity. To obtain sparse solutions, most previous works employed different regularizations such as l_1 and l_2 ones. However, they are limited in controlling sparsity level of solutions. In other words, the number of non-zero coefficients in solutions is unpredictable.

Unlike previous approaches, we have imposed a greedy algorithm, e.g., Frank-Wolfe algorithm, which can control severely the solution's sparsity. From Algorithms 2 and 3, the number of non-zero coefficients can be restricted by do not employing new latent components to optimize the cost function, when the number of non-zeros ones reaches to the limitation. Moreover, the preference of selecting the best latent components to optimize allows our algorithm to achieve more sparse solutions than other algorithms while keeping the optimality of solution.

C. Distributability and Parallelizability

From Algorithms 2 and 3, we have several remarks:

- Inference of data instances can be distributed undoubtedly over machines. This is important for designing distributed algorithms.
- Running time of inference depends mostly on finding the best latent component. Furthermore, computing the partial derivative for each latent component is separated totally. As a result, this computation can be paralleled. Hence, we can reduce effectively the responding time in real applications.



(a) Basic NMF (b) Simplicial NMF

Fig. 1. Latent components with $K = 25$ for digit database

Moreover, regarding to learning algorithms, the learning algorithm for KL-divergence is absolutely fast and can be distributed easily. For *Euclidean* distance, distributed algorithms in [14] can be employed.

VI. Experiments

This section investigates the effectiveness of our approach and algorithms for two divergence functions with *Euclidean* distance and *KL*-divergence by four criteria: interpretability, sparsity of solutions, performance in classification tasks and loss information measure. More particularly, our algorithm for *Euclidean* distance is compared to NMF [3], spNMF [17], oNMF[8], and cNMF [18], while the other one with *KL*-divergence is compared to kl-NMF[3], local non-negative matrix factorization (locNMF)[5], convolutional NMF(conNMF) [19], and Nonsmooth Nonnegative Matrix Factorization (nsNMF)[7]. The implemented codes are at http://www.ee.columbia.edu/ grindlay/code.html.

In this investigation, we use two typical databases of images and text. The digit dataset has 4000 random-selected samples from http://yann.lecun.com/exdb/mnist/, and 4327 labeled spam emails are all downloaded from http://csmining.org/index.php/spam-email-datasets-.html. For the email dataset, after normalizing data such as numbers turning into $number$ term and plural noun into single noun, we compute tf-idf (http://en.wikipedia.org/wiki/Tf-idf) for 32906 distinct terms as convenient features for data instances. We have compared our approach with other related approaches using *Euclidean* distance for the image dataset and KL-divergence for the text dataset. The obtained results are highly competitive.

A. Interpretation

In conducting experiments for the digit dataset, we have run with different parameters $K \in \{10, 15, 20, 25, 30, 35, 40\}$. We realize that approaches begin finding out part-based representation of data instances from $K = 25$. Figure VI-A shows latent components of NMF [2] and sNMF with $K = 25$ for the digit dataset. Obviously, latent components of sNMF are small part curves of digits, while ones of Lee 2001's NMF VI-A also gives a part-based representation but they are bigger curves. The result leads to conclusion that our approach found out a better part-based representation for data instances. Moreover, when factorizing matrices is comleted, sNMF's coefficients are in $[0, 1]$ with sum equal to 1, so they can represent the role of latent components in instances, while coefficients in other formulations does not. They are nonnegative numbers, which only represent the measure of basic vectors.

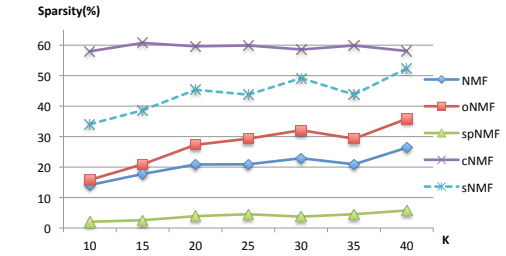


Fig. 2. Sparsity of new coefficients for *Euclidean* distance with $K = 30$

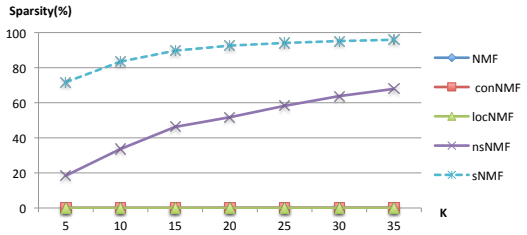


Fig. 3. Sparsity of new coefficients for KL-divergence with $K = 30$

B. Sparse Representation

In order to compare the sparsity of solutions, we compute the percentage of zero coefficients

$$\frac{number\ of\ zero\ coefficients}{number\ of\ coefficients} \times 100$$

The results are highly competitive with other methods. For *Euclidean* distance, although our algorithm's sparsity is only less than cNMF [18] (Figure 2), it has lower information loss and higher performance in classification. In addition, especially for *KL*-divergence, our approach retains the best sparse solutions (Figure 3), while it still has the best result for the other measures.

C. Performance for classification

Classification quality is one of measures that evaluates our method's effectiveness as NMF is often considered as a dimension redution technique used widely in classification. In this experiment, we use Random Forest[1], a robust algorithm for classification. Observing Figures 4 and 5, our method is one of methods with the lowest errors in testing. For *Euclidean* distance and the digit dataset, the result of our method is very close to the best method oNMF [8]. Meanwhile, for KL-divergence and spam dataset, our approach obtains the lowest misclassification with $K = 15$ and $K = 35$.

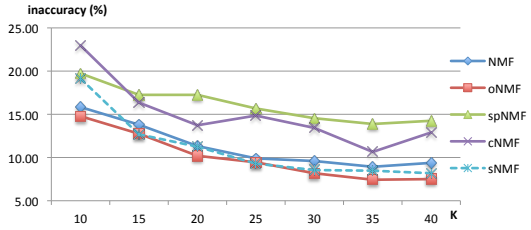


Fig. 4. Inaccuracy for Digit Classification

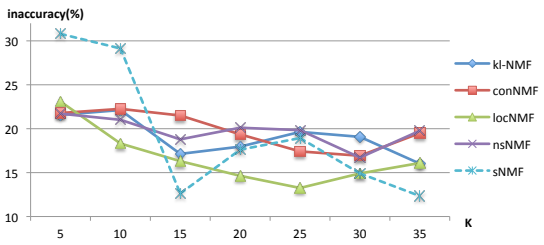


Fig. 5. Inaccuracy for Spam Classification

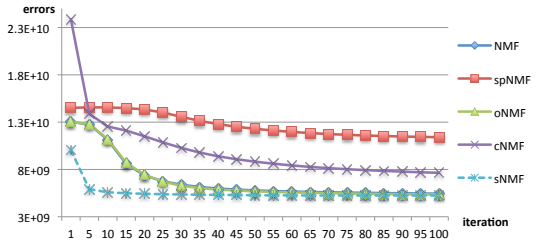


Fig. 6. Information Loss for Squared *Euclidean* Distance with $K = 30$

D. Information Loss and Convergence Speed

For dimension reduction, information loss criterion is one of the most important measure. From observing Figures 5 and 6, our approach has the lowest information loss.

In addition, convergence speed is a significant measure to evaluate updating algorithms because algorithms having more iterations require more computation and also time for loading data. We can realize easily from Figures 5 and 6 that our algorithm has the fast convergence speed while it gains the best optimized solutions with at the least number of iterations.

VII. Conclusion

In this paper, we have proposed a new formulation for NMF, simplicial nonnegative matrix factorization (sNMF). This formulation considers NMF as component analysis, in which each data instance is modeled as a convex combination of latent components. Two algorithms derived from Frank-Wolfe algorithm are designed for the two most popular divergence functions with *Euclidean* and KL-divergence, to control directly and effectively sparsity of solutions. We theoretically proved that our algorithms have low complexity in inference, sparsity, and distributability and parallelizability. Our experimental evaluation showed the effectiveness of our approach via significant criteria such as interpretability, sparsity, performance in classification task and information loss. Our obtained results are highly competitive with state-of-the-art approaches. In future research, we will deeper analyze our algorithms, apply them to other divergence functions and also conduct testing on other datasets to compare with other formualtions.

Acknowledgement

This work is partially sponsored by 322 Scholarship from Vietnam Ministry of Education and Training, and by Asian Office of Aerospace R&D under agreement number FA2386-13-1-4046.

[1] http://cran.r-project.org/web/packages/randomForest/

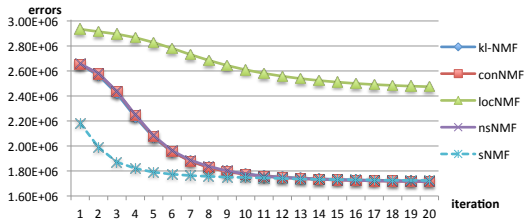


Fig. 7. Information Loss for $KL-$divergence with $K = 30$

References

[1] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 25, no. 6, pp. 1336–1353, 2013.

[2] 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.

[3] D. Seung and L. Lee, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, pp. 556–562, 2001.

[4] Z.-Y. Zhang, "Divergence functions of non negative matrix factorization: A comparison study," *Communications in Statistics-Simulation and Computation*, vol. 40, no. 10, pp. 1594–1612, 2011.

[5] S. Z. Li, X. W. Hou, H. J. Zhang, and Q. S. Cheng, "Learning spatially localized, parts-based representation," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1. IEEE, 2001, pp. I–207.

[6] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *J. Mach. Learn. Res.*, vol. 5, pp. 1457–1469, Dec. 2004.

[7] A. Pascual-Montano, J. M. Carazo, K. Kochi, D. Lehmann, and R. D. Pascual-Marqui, "Nonsmooth nonnegative matrix factorization (nsnmf)," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 3, pp. 403–415, 2006.

[8] S. Choi, "Algorithms for orthogonal nonnegative matrix factorization," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 2008, pp. 1828–1832.

[9] H. Li, T. Adal, W. Wang, D. Emge, and A. Cichocki, "Non-negative matrix factorization with orthogonality constraints and its application to raman spectroscopy," *The Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 48, no. 1-2, pp. 83–97, 2007.

[10] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari, *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. Wiley, 2009.

[11] K. L. Clarkson, "Coresets, sparse greedy approximation, and the frank-wolfe algorithm," *ACM Transactions on Algorithms (TALG)*, vol. 6, no. 4, p. 63, 2010.

[12] S. Boyd, "Alternating direction method of multipliers," in *Talk at NIPS Workshop on Optimization and Machine Learning*, 2011.

[13] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[14] C. Chu, S. K. Kim, Y.-A. Lin, Y. Yu, G. Bradski, A. Y. Ng, and K. Olukotun, "Map-reduce for machine learning on multicore," *Advances in neural information processing systems*, vol. 19, p. 281, 2007.

[15] C. L. Lawson and R. J. Hanson, *Solving least squares problems*. SIAM, 1974, vol. 161.

[16] Z. Yang, H. Zhang, Z. Yuan, and E. Oja, "Kullback-leibler divergence for nonnegative matrix factorization," *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 250–257, 2011.

[17] M. N. Schmidt, J. Larsen, and F.-T. Hsiao, "Wind noise reduction using non-negative sparse coding," in *Machine Learning for Signal Processing, 2007 IEEE Workshop on*. IEEE, 2007, pp. 431–436.

[18] C. Ding, T. Li, and M. Jordan, "Convex and semi-nonnegative matrix factorizations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 1, pp. 45–55, 2010.

[19] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *Independent Component Analysis and Blind Signal Separation*. Springer, 2004, pp. 494–499.