Kullback-Leibler Divergence for Nonnegative Matrix Factorization

Zhirong Yang, He Zhang, Zhijian Yuan, and Erkki Oja

Department of Information and Computer Science*
Aalto University, P.O.Box 15400, FI-00076, Espoo, Finland,
{zhirong.yang,he.zhang,zhijian.yuan,erkki.oja}@aalto.fi

Abstract. The I-divergence or unnormalized generalization of Kullback-Leibler (KL) divergence is commonly used in Nonnegative Matrix Factorization (NMF). This divergence has the drawback that its gradients with respect to the factorizing matrices depend heavily on the scales of the matrices, and learning the scales in gradient-descent optimization may require many iterations. This is often handled by explicit normalization of one of the matrices, but this step may actually increase the I-divergence and is not included in the NMF monotonicity proof. A simple remedy that we study here is to normalize the input data. Such normalization allows the replacement of the I-divergence with the original KL-divergence for NMF and its variants. We show that using KL-divergence takes the normalization structure into account in a very natural way and brings improvements for nonnegative matrix factorizations: the gradients of the normalized KL-divergence are well-scaled and thus lead to a new projected gradient method for NMF which runs faster or yields better approximation than three other widely used NMF algorithms.

1 Introduction

Nonnegative Matrix Factorization (NMF) is a powerful tool for signal processing, data analysis, and machine learning, that has attracted much research effort recently. The problem was first introduced by Paatero and Tapper [1]. After Lee and Seung [2] presented multiplicative update algorithms for NMF, a multitude of variants of NMF (see e.g. [3] for a survey) have been proposed. Most of these methods can be divided into two categories according to the approximation criterion: least square error or information divergence. For the latter category, the *generalized Kullback-Leibler divergence* or *I-divergence* used in Lee and Seung's algorithm is widely adopted in present applications.

In spite of a number of generalizations (see e.g. [4, 5, 3]), little research has been devoted to investigating the difference between I-divergence and the original Kullback-Leibler (KL) divergence. Actually the I-divergence difference measure has a number of drawbacks. Firstly, in many applications the data matrix can be

* Supported by the Academy of Finland in the project *Finnish Centre of Excellence in Adaptive Informatics Research.*

normalized before input to divergence-based NMF algorithms. This is the case when the relative differences among matrix entries are more important than their individual magnitudes. The normalization structure can provide valuable information for the NMF learning, but I-divergence neglects such information. Secondly, the Poisson noise model that underlies the I-divergence [2] is a discrete distribution defined only for nonnegative integers [6]. Thirdly, the gradients that provide critical information for the updating direction in optimization heavily depend on the scales of factorizing matrices, whose correct values are unknown beforehand. The additive gradient-based optimization of NMF with I-divergence can be very slow because it requires many iterations to recover from wrong initial scales.

In this paper we study the replacement of the I-divergence for NMF with the original KL-divergence for normalized data. Optimizing the new objectives is not trivial, where we are facing the challenge that the KL-divergence is non-separable over the matrix elements. Actually it belongs to the family of γ-divergence (see e.g. [3, Chapter 2]) whose optimization is unseen before. A new projected gradient method is then proposed for NMF based on normalized KL-divergence, which runs faster than two other additive optimization approaches and gives better approximation than the conventional multiplicative updates.

The rest of the paper is organized as follows. We briefly review the NMF problem based on the I-divergence in Section 2. In Section 3, we present NMF based on the normalized KL-divergence, including their objectives and corresponding optimization algorithms. Section 4 shows the empirical results which demonstrate the advantages of using normalized KL-divergence. The conclusions and future work are given in Section 5.

2 NMF based on I-Divergence

Given a nonnegative input data matrix $\mathbf{X} \in \mathbb{R}_+^{m\times n}$, *Nonnegative Matrix Factorization* (NMF) seeks a decomposition of $\mathbf{X}$ that is of the form $\mathbf{X} \approx \mathbf{WH}$, where $\mathbf{W} \in \mathbb{R}_+^{m\times r}$ and $\mathbf{H} \in \mathbb{R}_+^{r\times n}$ with the rank $r < \min(m, n)$. The matrix $\widehat{\mathbf{X}} = \mathbf{WH}$ is called the unnormalized approximating matrix of $\mathbf{X}$.

In previous work, the approximation has widely been achieved by minimizing one of the two measures: (1) the least square criterion $\varepsilon = \sum_{i,j}(X_{ij} - \widehat{X}_{ij})^2$ and (2) the *generalized Kullback-Leibler divergence* (or *I-divergence*)

$$D_I\left(\mathbf{X}||\widehat{\mathbf{X}}\right) = \sum_{ij}\left(X_{ij}\log\frac{X_{ij}}{\widehat{X}_{ij}} - X_{ij} + \widehat{X}_{ij}\right). \tag{1}$$

In this paper we focus on the second approximation criterion, which is particularly useful for sparse counting data of small occurrences. In what follows, we call NMF based on the I-divergence I-NMF, to distinguish it from the one based on the original KL-divergence described below.

3 NMF based on KL-Divergence

In many applications, the input data matrix is or can be normalized before its nonnegative matrix factorization. The normalization scheme can provide valuable information for NMF algorithms and should be taken into account. This motivates us to improve NMF by using the original or normalized KL-divergence.

3.1 Normalized Kullback-Leibler Divergence

Let $\mathbf{X}$ denote the normalized input data matrix. According to the original KL-divergence definition, the NMF approximation should be $\mathbf{X} \approx \widetilde{\mathbf{X}}$ by the criterion

$$D_{KL}\left(\mathbf{X}||\widetilde{\mathbf{X}}\right) = \sum_{i,j} X_{ij} \log \frac{X_{ij}}{\widetilde{X}_{ij}}, \tag{2}$$

where $\widetilde{\mathbf{X}}$ is obtained from $\widehat{\mathbf{X}} = \mathbf{WH}$ using the same normalization scheme that was used for $\mathbf{X}$. Common normalization schemes include

- matrix-wise normalization: $\sum_{ij} X_{ij} = 1$. Then $\widetilde{X}_{ij} = \widehat{X}_{ij} / \sum_{ab} \widehat{X}_{ab}$;
- row-wise normalization: for all i, $\sum_j X_{ij} = 1$. Then $\widetilde{X}_{ij} = \widehat{X}_{ij} / \sum_b \widehat{X}_{ib}$;
- column-wise normalization: for all j, $\sum_i X_{ij} = 1$. Then $\widetilde{X}_{ij} = \widehat{X}_{ij} / \sum_a \widehat{X}_{aj}$.

The following derivations will focus on the matrix case to avoid notational clutter, but the discussions can easily be extended to the other two cases. The empirical advantages of row-wise normalized KL-NMF is shown in Section 4.

Normalized data matrices exist widely in applications. A matrix-wise normalization example is to approximate a symmetric affinity matrix [7]. For row-wise or column-wise normalization, a good example is the document-term occurrence matrix commonly used in information retrieval.

We choose the original KL-divergence also because it can bring us better stability. Let us take the matrix-wise normalization for example. Writing out (2), one can see that up to a constant the KL-divergence

$$D_{KL}^{\text{mat}}\left(\mathbf{X}||\widetilde{\mathbf{X}}\right) = \sum_{ij} X_{ij} \log \frac{X_{ij}}{\widehat{X}_{ij}} + \log \sum_{ij} \widehat{X}_{ij} \tag{3}$$

differs from the I-divergence with a logarithm before $\sum_{ij} \widehat{X}_{ij}$. As we shall see, this logarithm plays a key role in efficiently adjusting the scale of $\widehat{\mathbf{X}}$ for both additive and multiplicative optimization.

It has recently been shown that at the stationary points the I-NMF also preserves the column-wise and row-wise sums of the input matrix [8]. However, so far there is no optimization algorithm that theoretically guarantees to achieve such exact stationary points. In practice, the learning with I-divergence can still be inconsistent with the normalization of input matrix.

3.2 Equivalence to pLSI

It has been shown that I-NMF is equivalent to the *Probabilistic Latent Semantic Indexing* (pLSI) under certain conditions [9]. Actually NMF based on KL-divergence (KL-NMF) has closer relationship to pLSI than I-NMF. KL-NMF optimizes exactly the same objective as pLSI by its definition. The requirement of unitary column sum in pLSI can be fulfilled by applying column normalization only once after the iterative learning.

The I-divergence is separable over the matrix elements, but pLSI requires unitary sum of the input matrix and of its approximate. I-divergence belongs to the family of α- or β-divergences, while pLSI and KL-divergence belong to the family of γ-divergences which are non-separable (see e.g. [3, Chapter 2]).

In I-NMF, the equivalence to pLSI can be enforced by employing column normalization as shown by Proposition 2 in [9]. However, the extra normalization steps are not included in the convergence proof. The normalization step itself can indeed often violate the monotonic decrease of the objective. By contrast, both additive and multiplicative algorithms for nonnegative matrix factorizations based on KL-divergence do not require such an extra normalization step, which facilitates their convergence analysis.

3.3 Projected gradient algorithms for NMF

The most popular solution for I-NMF is alternatively applying two multiplicative update rules [10]. Such EM-like multiplicative algorithms for NMF do not require user-specified optimization parameters and thus are widely used. However, Gonzales and Zhang [11] as well as Lin [12] found that the monotonicity guaranteed by the proof of multiplicative updates may not imply the full Karush-Kuhn-Tucker conditions. Therefore, it remains possible to find a better objective by using some other optimization methods instead of multiplicative updates. In addition, multiplicative updates are often slower in the long-run training compared with gradient approaches such as [13]. This also motivates the use of additive updates based on the gradient information.

The most commonly used additive approach for I-NMF is the *Projected Gradients* [13], where the new estimate is obtained by first calculating the unconstrained steepest-descent update and then zeroing its negative elements. Lin [13] employed a line search method with the Armijo rule for selecting the learning step size η. Their method alternates the minimization over either $\mathbf{W}$ or $\mathbf{H}$, with the other matrix fixed. Denote $f(\mathbf{W}, \mathbf{H})$ the NMF objective, i.e. I-divergence or KL-divergence in this paper. When minimizing f over $\mathbf{H}$, the Armijo's rule tries to find the largest η that satisfies the *sufficient decrease condition*

$$f(\mathbf{W}, \mathbf{H}^{\text{new}}) - f(\mathbf{W}, \mathbf{H}) \leq \sigma \text{Tr}\left(\nabla_{\mathbf{H}} f(\mathbf{W}, \mathbf{H})(\mathbf{H}^{\text{new}} - \mathbf{H})^T\right), \tag{4}$$

where $0 < \sigma < 1$. The concrete form of the gradient $\nabla_{\mathbf{H}} f$, depending on the divergence used, is given by Eqs (5) and (7) below. Assuming that η does not vary too much in consecutive iterations, the improved minimization is described

Algorithm 1 Projected gradients with Armijo's rule

Initialize $\mathbf{H}$. Set $\eta = 1$.
for $i = 1$ to k **do**
  if η satisfies Eq. (4) **then**
    repeatedly increase it by $\eta \leftarrow \eta/\rho$ until either ρ does not satisfy Eq. (4) or $\mathbf{H}(\eta/\rho) = \mathbf{H}(\eta)$
  else
    repeatedly decrease η by $\eta \leftarrow \eta \cdot \rho$ until η satisfies Eq. (4)
  end if
  Set $\mathbf{H}^{\text{new}} = \max(\mathbf{0}, \mathbf{H} - \eta \nabla_{\mathbf{H}} f(\mathbf{W}, \mathbf{H}))$.
end for

in Algorithm 1 [13], in which ρ is the dilation/shrinkage base for line search. The same algorithm applies to the minimization over $\mathbf{W}$ with $\mathbf{H}$ fixed.

Although a number of projected gradient algorithms (e.g. [14, 15, 3], Chapter. 4-6) have been proposed for NMF based on least squared errors, the speedup advantage is more difficult to obtain for the approximation based on the I-divergence. The major reason is that the gradients of I-NMF

$$\nabla_{\mathbf{H}} D_I(\mathbf{X}||\mathbf{WH}) = -\mathbf{W}^T\mathbf{Z} + \mathbf{W}^T\bar{\mathbf{Z}} \tag{5}$$

$$\nabla_{\mathbf{W}} D_I(\mathbf{X}||\mathbf{WH}) = -\mathbf{Z}\mathbf{H}^T + \bar{\mathbf{Z}}\mathbf{H}^T \tag{6}$$

heavily depend on the scaling of $\mathbf{W}$ and $\mathbf{H}$, where $\mathbf{Z}$ and $\bar{\mathbf{Z}}$ are of size $m \times n$ with $Z_{ij} = X_{ij}/(\mathbf{WH})_{ij}$ and $\bar{Z}_{ij} = 1$. For example, if the entries of initial $\mathbf{W}$ are overly large, the second term in Eq. (5) will dominate the gradient because in the first term the scale of $\mathbf{W}$ cancels out. Unlike multiplicative updates which can remedy for an improper scale by alternation between $\mathbf{W}$ and $\mathbf{H}$, the projected gradient algorithm requires many more iterations to recover from such a wrong guess of scales. This is especially problematic for large-scale factorization tasks. The badly scaled gradients also make the second order optimization methods such as Newton or quasi-Newton algorithms ill-posed and even fail to converge, as shown in Section 4.

By contrast, the normalized Kullback-Leibler divergence does not suffer from such a scaling problem. Consider first matrix-wise normalization. Then the logarithm in the second term of Eq. (3) leads to an inverse normalization factor $\alpha = \sum_{ab} (\mathbf{WH})_{ab}$ in the gradients:

$$\nabla_{\mathbf{H}} D_{KL}^{\text{mat}}(\mathbf{X}||\widetilde{\mathbf{X}}) = -\mathbf{W}^T\mathbf{Z} + \frac{1}{\alpha}\mathbf{W}^T\bar{\mathbf{Z}} \tag{7}$$

$$\nabla_{\mathbf{W}} D_{KL}^{\text{mat}}(\mathbf{X}||\widetilde{\mathbf{X}}) = -\mathbf{Z}\mathbf{H}^T + \frac{1}{\alpha}\bar{\mathbf{Z}}\mathbf{H}^T. \tag{8}$$

Such normalization factors can automatically stabilize gradient-based optimization. The learning can thus focus on adjusting the relative values among entries of factorizing matrices, which results in more efficient algorithms.

4 Experiments

The normalized KL-divergence for NMF leads to well-scaled gradients which favor stable additive optimization approaches. We have compared the projected gradient method using the Armijo rule based on the gradients of I-divergence and row-normalized KL-divergence, as well as the multiplicative I-NMF algorithm [2] and a quadratic programming method based on I-divergence [3]. We refer to the four compared methods as *I-Armijo*, *KL-Armijo*, *I-multiplicative*, and *I-quadratic*, respectively.

We have used four datasets *med*[1] (1033×5831), *cran*[1] (1398×4612), *cisi*[1] (1460×5609), and *webkb4*[2] (4193×1000). The document-term matrices are preprocessed according to the TF-IDF weighting scheme, which is widely used in information retrieval and text mining, and then normalized to unit row-sum. We selected these datasets because (1) our contribution addresses the normalization structure of input data and (2) the Armijo rule can be efficiently performed for large-scale but sparse matrices.

It is important to notice that $D_{KL}(\mathbf{X}||\widetilde{\mathbf{X}}) = D_I(\mathbf{X}||\widetilde{\mathbf{X}})$ if both $\mathbf{X}$ and $\widetilde{\mathbf{X}}$ are normalized in the same scheme. On the other hand, generally $D_I(\mathbf{X}||\widetilde{\mathbf{X}}) \neq D_I(\mathbf{X}||\widehat{\mathbf{X}})$. This enables us to compare the approximation performance of the four selected methods for normalized matrices using $D_I(\mathbf{X}||\widetilde{\mathbf{X}})$.

Following [13], we set the line search base ρ to 0.1. The factor σ for determining sufficient decrease is set to 10^{-5} in our experiments. Each method terminates if maximal time (3600 seconds) or maximal number of iterations (1000, 1000, 10000, 100, respectively) is reached. Maximal ten iterations for inner loops have been used in I-Armijo and KL-Armijo. We repeated 50 times for every method on each dataset and recorded their resulting I-divergences and KL-divergences.

Figure 1 shows the evolution curves of I-divergences versus learning time. The I-quadratic method violates the monotonicity quite often and seems to diverge in the experiment. Another projected gradient method based on the I-divergence, I-Armijo, requires about ten minutes to decrease the objective below 10^4 and is then stuck around 3800. The objectives using the other two methods, I-multiplicative and KL-Armijo, become smaller than 3400 in less than two minutes. The multiplicative algorithm seems faster in early iterations but gives little improvement after one minute, which ends up with the objective value 3378. By contrast, the proposed projected gradient method based on KL-divergence steadily minimizes the objective until it becomes stable at 3314 after about four minutes.

More extensive results are shown in Table 1. The two methods using projected gradients of I-divergence perform poorly in terms of both objectives. Their resulting mean divergences are much higher than the other two approaches. In practice, we find that I-quadratic often yields extremely bad results, which leads to drastically high variance across different tries. I-multiplicative and KL-Armijo are free of such instability, where they both converge with reasonable divergences.

[1] `http://www.cs.utk.edu/~lsi/`

[2] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/`

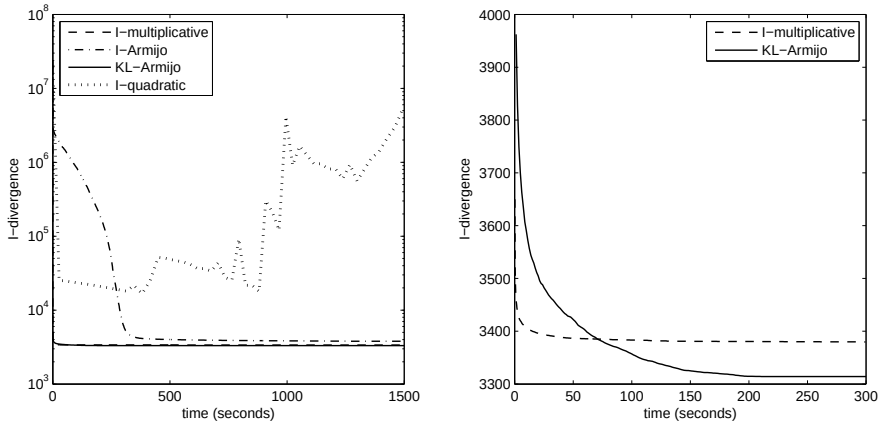

Fig. 1. (Left) I-divergence evolutions using the four compared methods on the *med* dataset. (Right) The zoom-in comparison between I-multiplicative and KL-Armijo.

Table 1. I-divergences shown in format $\mu \pm \sigma$, where μ is the mean and σ the standard deviation. Boldface table cells contain the smallest mean divergences.

	I-multiplicative	I-Armijo	KL-Armijo	I-quadratic
med:	3376.11 ± 15.34	3915.41 ± 117.25	$\mathbf{3337.97 \pm 21.25}$	$1.7 \times 10^{14} \pm 4.2 \times 10^{14}$
cran:	5008.25 ± 12.13	5555.26 ± 85.49	$\mathbf{4961.89 \pm 12.43}$	$4.0 \times 10^{17} \pm 1.1 \times 10^{18}$
cisi:	4058.06 ± 7.43	4697.35 ± 129.76	$\mathbf{4023.99 \pm 12.64}$	$6.4 \times 10^{15} \pm 9.1 \times 10^{15}$
webkb4:	9145.38 ± 42.81	10528.07 ± 367.50	$\mathbf{9113.02 \pm 43.28}$	$1.9 \times 10^{7} \pm 1.3 \times 10^{7}$

By contrast, the latter using KL-divergence and projected gradient descent can find even better objectives. This is probably because monotonicity guaranteed by multiplicative updates may not imply the full KKT condition.

5 Conclusions

We have studied the Kullback-Leibler divergence versus I-divergence in NMF for normalized input data. Using the gradients of the former results in a faster additive optimization algorithm that yields better approximation than three other existing methods. Actually, both theoretical and practical advantages indicate that there would be good reasons to replace the I-divergence with KL-divergence for NMF and its variants.

More advanced optimization algorithms beyond the simple Armijo rule, for example, conjugate gradient descent, could be constructed by using the proposed gradients that provide better learning directions. In addition, the improvement towards uniqueness of KL-NMF needs further investigation. Appropriate constraints or priors could significantly reduce ambiguity between factorizing matrices. For extensions, the proposed method can readily include penalizations

such as additional L1 or L2 norms of the factorizing matrices. Other extensions such as the use of Automatic Relevance Determination to automatically select the low rank in approximation can also be implemented as in conventional NMF.

References

1. Paatero, P., Tapper, U.: Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. Environmetrics **5** (1994) 111–126
2. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401** (1999) 788–791
3. Cichocki, A., Zdunek, R., Phan, A.H., Amari, S.: Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis. John Wiley (2009)
4. Dhillon, I.S., Sra, S.: Generalized nonnegative matrix approximations with bregman divergences. In: Advances in Neural Information Processing Systems. Volume 18. (2006) 283–290
5. Févotte, C., Bertin, N., Durrieu, J.L.: Nonnegative matrix factorization with the Itakura-Saito divergence: With application to music analysis. Neural Computation **21**(3) (2009) 793–830
6. Gullberg, J.: Mathematics: From the Birth of Numbers. W. W. Norton & Company (1997)
7. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. Journal of Machine Learning Research **9** (2008) 2579–2605
8. Ho, N.D., Dooren, P.V.: Non-negative matrix factorization with fixed row and column sums. Linear Algebra and its Applications **429**(5-6) (2008) 1020–1025
9. Ding, C., Li, T., Peng, W.: On the equivalence between non-negative matrix factorization and probabilistic laten semantic indexing. Computational Statistics and Data Analysis **52**(8) (2008) 3913–3927
10. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. Advances in Neural Information Processing Systems **13** (2001) 556–562
11. Gonzales, E.F., Zhang, Y.: Accelerating the lee-seung algorithm for non-negative matrix factorization. Technical report, Dept. of Computational and Applied Mathematics, Rice University (2005)
12. Lin, C.J.: On the convergence of multiplicative update algorithms for nonnegative matrix factorization. IEEE Transactions on Neural Networks **18**(6) (2007) 1589–1596
13. Lin, C.J.: Projected gradient methods for non-negative matrix factorization. Neural Computation **19** (2007) 2756–2779
14. Kim, D., Sra, S., Dhillon, I.S.: Fast projection-based methods for the least squares nonnegative matrix approximation problem. Statistical Analysis and Data Mining **1**(1) (2008) 38–51
15. Kim, H., Park, H.: Nonnegative matrix factorization based on alternating nonnegativity-constrained least squares and the active set method. SIAM Journal on Matrix Analysis and Applications **30**(2) (2008) 713–730