

A HYBRID ITERATIVE ALGORITHM FOR NONNEGATIVE MATRIX FACTORIZATION

Ştefan M. Şoltuz, Wenwu Wang and Philip J.B. Jackson

Centre for Vision Speech and Signal Processing
Dept. of Electronic Engineering, University of Surrey, Guildford, U.K.
Emails:[ssoltuz; w.wang; p.jackson]@surrey.ac.uk

ABSTRACT

The aim of Non-negative Matrix Factorization (NMF) is to decompose a non-negative matrix into a product of two (or multiple) non-negative matrices with reduced ranks. Several iterative methods have been developed for this purpose, e.g. the Alternating Least Squares (ALS) or Lee-Seung (LS) multiplicative methods. Despite its fast convergence, the ALS algorithm suffers from its instability, and may diverge in practice. The LS method, although reasonably stable, is known to converge slowly. In this paper, we develop a hybrid algorithm using mixed iterations based on these two methods. We show theoretically that the hybrid algorithm outperforms both methods by achieving a better tradeoff between the convergence speed and stability without increasing computational complexity. We also provide numerical examples in which we compare our hybrid algorithm with the LS and ALS algorithms.

Index Terms— Non-negative matrix factorization (NMF), Mixed iterations, Alternating least squares, Lee-Seung method

1. INTRODUCTION

Non-negative matrix factorization (NMF) attempts to decompose a non-negative data matrix into a product of two (or multiple) non-negative matrices [6]. This technique has found many applications in, for example, source separation, dimensionality reduction and clustering. There are several other methods that can deal with similar problems, e.g. principal component analysis (PCA) and singular value decomposition (SVD). One major problem with PCA is that the basis vectors may have both positive and negative components, and the data are represented as linear combinations of these vectors with positive and negative coefficients. In many applications, the negative components contradict physical realities. For example, the pixels in a gray-scale image have non-negative intensities, so an image with negative intensities cannot be reasonably interpreted. To address this problem, NMF was proposed to search for a representative basis with only nonnegative vectors, see [2]. In contrast to SVD which has a unique factorization and the orthogonality property, NMF has advantages in

This work was supported in part by the EPSRC of the UK. The first author is also with T. Popoviciu, Institute of Numerical Analysis, Cluj, Romania.

the properties of sparsity, nonnegativity, and interpretability [1].

To find an appropriate NMF decomposition, a cost function based on, e.g. the squared Euclidean distance, is usually defined to measure the estimation error between the original non-negative matrix and its decomposed product. Several iterative methods have been proposed for the optimisation of such a cost function. The most widely used method is perhaps the multiplicative algorithm by Lee-Seung (LS) [5] [6], for which a modified form of iterations has been provided by Lin [8], and some theoretical results are also given in [4]. The LS algorithm has good convergence stability and also maintains the non-negativity of the decomposed matrices during iterations. However, it is also well-known for its slow convergence. Alternating least squares (ALS) method is also used for the same purpose by fixing one argument and finding the least-squares solution for the other in an alternating manner [1]. The ALS algorithm converges much faster as compared with LS algorithm, however, it is not stable and may diverge frequently in practice. In this work, we first analyse the convergence issues of these two algorithms and then develop a hybrid method based on their complementarity in Section 2. The convergence performance of the proposed method is analysed theoretically in Section 3 and also compared numerically with LS and ALS algorithms in Section 4, followed by the concluding remarks in Section 5.

2. THE PROPOSED METHOD

2.1. The optimisation criterion

For a given matrix \mathbf{V} , NMF finds \mathbf{W} and \mathbf{H} such that:

$$\mathbf{V} = \mathbf{WH}, \quad (1)$$

where $\mathbf{V} \in \mathbb{R}_+^{m \times p}$, $\mathbf{W} \in \mathbb{R}_+^{m \times r}$ and $\mathbf{H} \in \mathbb{R}_+^{r \times p}$. To find such a decomposition the following cost function is usually considered [5]

$$F(\mathbf{W}, \mathbf{H}) = \frac{1}{2} \|\mathbf{V} - \mathbf{WH}\|_F^2, \quad (2)$$

where $\|\cdot\|_F$ is the Frobenious norm.

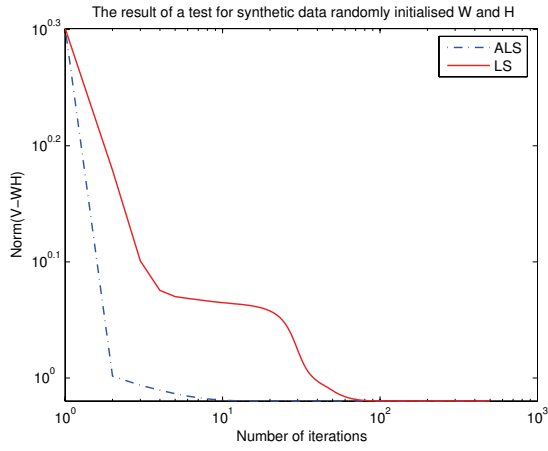


Fig. 1. Convergence curve of a random test where both the ALS and the LS algorithm converge.

2.2. The LS and ALS iterations

The LS algorithm uses the following multiplicative iterations to minimise function (2):

$$\mathbf{H}_{n+1} = \mathbf{H}_n \odot \frac{(\mathbf{W}_n^T \mathbf{V})}{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n)}, \quad (3)$$

$$\mathbf{W}_{n+1} = \mathbf{W}_n \odot \frac{(\mathbf{V} \mathbf{H}_{n+1}^T)}{(\mathbf{W}_n \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T)}, \quad (4)$$

where \odot denotes the element-wise matrix multiplication, n is the iteration number, T is a matrix transpose, and the divisions operate element-wise. In practice, a *small* positive quantity (e.g. $\varepsilon = 10^{-9}$), is usually added to the denominators in the above updates to prevent the numerator from being divided by zero.

$$\mathbf{H}_{n+1} = \mathbf{H}_n \odot \frac{(\mathbf{W}_n^T \mathbf{V})}{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n) + \varepsilon}, \quad (5)$$

$$\mathbf{W}_{n+1} = \mathbf{W}_n \odot \frac{(\mathbf{V} \mathbf{H}_{n+1}^T)}{(\mathbf{W}_n \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T) + \varepsilon}. \quad (6)$$

Adding ε provides potentially more stable iterations and make the iterations theoretically more tractable [8].

The ALS algorithm minimises function (2) based on the following iterations,

$$\mathbf{H}_{n+1} = (\mathbf{W}_n^T \mathbf{W}_n)^{-1} (\mathbf{W}_n^T \mathbf{V}), \quad (7)$$

$$\mathbf{W}_{n+1} = \mathbf{V} \mathbf{H}_{n+1}^T (\mathbf{H}_{n+1} \mathbf{H}_{n+1}^T)^{-1}. \quad (8)$$

The non-negativity is not guaranteed with such iterations due to the matrix inverse. In order to mitigate this problem, the factorized matrices are further projected onto the nonnegative orthant as follows,

$$\mathbf{H}_{n+1} = P_+(\mathbf{H}_{n+1}), \quad (9)$$

$$\mathbf{W}_{n+1} = P_+(\mathbf{W}_{n+1}), \quad (10)$$

where the projection $P_+[\cdot]$ is defined as $P_+[\mathbf{A}] = \max(\mathbf{A}, \mathbf{0})$, and $\mathbf{0}$ is a zero-filled matrix having the same dimension as \mathbf{A} , and \max is an operator comparing two matrices element-wise taking the elements with greater value.

2.3. Convergence issues of the LS and ALS algorithms

As reported by several studies, e.g. [9] and [7], iterations (9) and (10) converge (when they do) faster than (3) and (4) or (5) and (6). However, as observed in our experiments, the convergence performance of ALS is not consistent and may easily diverge. The convergence behavior can be observed in the following two random tests, shown in Figures 1 and 2, for an artificially generated data matrix $\mathbf{V} \in \mathbb{R}_+^{3 \times 4}$. In both tests, r was set to 2 and $\mathbf{W} \in \mathbb{R}_+^{3 \times 2}$ and $\mathbf{H} \in \mathbb{R}_+^{2 \times 4}$ were randomly initialised. Figure 1 indicates that the ALS algorithm has faster convergence rate than the LS algorithm when both algorithms converge. However, Figure 2 shows that for another test, the ALS algorithm may diverge while the LS algorithm still converges nicely.

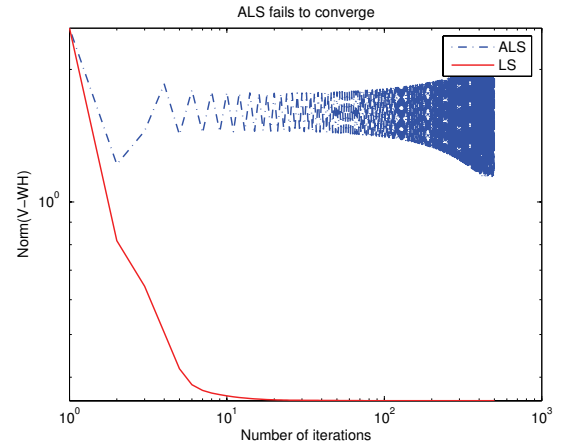


Fig. 2. Convergence curve of a random test where the ALS algorithm diverges while the LS algorithm still converges.

2.4. The mixed iterations

To mitigate the convergence issues discussed above, we develop a hybrid method in this section. Essentially, using the combination of the iterations from the ALS and LS algorithms, the hybrid algorithm achieves a tradeoff between the convergence rate and the stability of the considered algorithms. It can be observed that the iterations in both the LS and ALS methods take the following general form:

$$\mathbf{H}_{n+1} = \mathbf{H}_n + \mathbf{A}_n^{-1} (\mathbf{W}_n^T \mathbf{V} - \mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n), \quad (11)$$

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \mathbf{B}_n^{-1} (\mathbf{V} \mathbf{H}_{n+1}^T - \mathbf{W}_n \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T), \quad (12)$$

where \mathbf{A}_n and \mathbf{B}_n are the step-size regulating matrices at iteration n . Specifically, in LS algorithm [5], these matrices are

updated per element by

$$(\mathbf{A}_n)_{ab} = \delta_{ab} \left(\frac{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n)_{ab}}{(\mathbf{H}_n)_{ab}} \right), \quad (13)$$

$$(\mathbf{B}_n)_{ab} = \delta_{ab} \left(\frac{(\mathbf{W}_n \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T)_{ab}}{(\mathbf{W}_n)_{ab}} \right), \quad (14)$$

where $\delta_{ab}(\cdot)$ is the Kronecker symbol and ab is the index of an element in a matrix in row a and column b . Since both \mathbf{A}_n and \mathbf{B}_n are diagonal matrices, the inverse will be simply the matrices with reciprocal diagonal elements. If \mathbf{A}_n and \mathbf{B}_n are set as follows,

$$\mathbf{A}_n = \mathbf{W}_n^T \mathbf{W}_n, \quad (15)$$

$$\mathbf{B}_n = \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T, \quad (16)$$

we can obtain iterations (7) and (8) from (11) and (12), which lead to the ALS algorithm. The convergence for such iterations was reported in [3]. We remark that the quantity used in (15) and (16) is similar to the projection operator for the well-known least-squares method.

To improve the convergence speed of (3) and (5) and the stability of (9) and (10), we adopt the following mixed forms for \mathbf{A}_n and \mathbf{B}_n in (11) and (12),

$$(\mathbf{A}_n)_{ab} = \delta_{ab} \left(\frac{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n)_{ab} + \varepsilon}{(\mathbf{H}_n)_{ab}} \right) \quad (17)$$

$$\mathbf{B}_n = \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T. \quad (18)$$

As a result, we obtain the hybrid algorithm as follows

$$\mathbf{H}_{n+1} = \mathbf{H}_n \odot \frac{(\mathbf{W}_n^T \mathbf{V})}{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n) + \varepsilon}, \quad (19)$$

$$\mathbf{W}_{n+1} = \mathbf{V} \mathbf{H}_{n+1} (\mathbf{H}_{n+1} \mathbf{H}_{n+1}^T)^{-1}. \quad (20)$$

Although the algorithm is derived intuitively, its improved convergence performance over the ALS and LS algorithms can be justified both analytically and numerically, as discussed in subsequent sections.

3. ANALYTICAL RESULTS

3.1. The convergence

Definition 1. [5] $G(\mathbf{h}, \mathbf{h}')$ is an auxiliary function for $F(\mathbf{h})$ if the conditions (i) $G(\mathbf{h}, \mathbf{h}') \geq F(\mathbf{h})$, (ii) $G(\mathbf{h}, \mathbf{h}') = F(\mathbf{h})$ are satisfied.

Similar to the Taylor expansion of (2) where \mathbf{A}_n takes the form of $\mathbf{W}_n^T \mathbf{W}_n$, the auxiliary function for (5) is given by,

$$G(\mathbf{h}, \mathbf{h}_n) = F(\mathbf{h}_n) + (\mathbf{h} - \mathbf{h}_n)^T \nabla_{\mathbf{h}} F(\mathbf{h}_n) \quad (21)$$

$$+ \frac{1}{2} (\mathbf{h} - \mathbf{h}_n)^T (\mathbf{A}_n) (\mathbf{h} - \mathbf{h}_n). \quad (22)$$

Recall Lemma 2 from [5] or Lemma 1 from [8], we see that $\mathbf{A}_n - \mathbf{W}_n^T \mathbf{W}_n$ is a positive definite matrix. In [8] it was noted that if \mathbf{A}_n is given by (13), then iteration (5) for the update of \mathbf{H}_n converges. However, for the update of \mathbf{W}_n , it was not required that $\mathbf{B}_n - \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T$ be a strictly positive matrix. Therefore, we choose $\mathbf{B}_n = \mathbf{H}_{n+1} \mathbf{H}_{n+1}^T$. Consequently, using the following Lemma, it is straightforward to observe that if iterations (5) and (6) converge, then iterations (19) and (20) converge too.

Lemma 2. Suppose \mathbf{H}_n and \mathbf{W}_n are given by (19) and (20) respectively. If $\mathbf{H}_n \rightarrow \mathbf{H}^*$, then $\mathbf{W}_n \rightarrow \mathbf{W}^*$, where \mathbf{W}^* is the limit of (5).

Proof. Since $\mathbf{V} = \mathbf{W}^* \mathbf{H}^*$, multiply with \mathbf{H}^{*T} , and then multiply with the inverse $(\mathbf{H}^* \mathbf{H}^{*T})^{-1}$, to obtain

$$\mathbf{W}^* = \mathbf{V} \mathbf{H}^{*T} (\mathbf{H}^* \mathbf{H}^{*T})^{-1}. \quad (23)$$

Suppose that from an iteration n we have $\mathbf{H}_n \approx \mathbf{H}^*$. From (20), we know that

$$\mathbf{W}_{n+1} = \mathbf{V} \mathbf{H}_{n+1}^T (\mathbf{H}_{n+1} \mathbf{H}_{n+1}^T)^{-1} \quad (24)$$

$$\approx \mathbf{V} \mathbf{H}^{*T} (\mathbf{H}^* \mathbf{H}^{*T})^{-1}. \quad (25)$$

Therefore, we obtain $\mathbf{W}_{n+1} \approx \mathbf{W}^*$. \square

3.2. Stability and Convergence Rate

The proposed hybrid algorithm is more stable than the ALS algorithm, as shown in the following analysis. Suppose we take $\|\cdot\|$ as the sup norm. Without loss of generality, we assume $\|\mathbf{V}\| = 1$. We shall compare the stability of iterations (7) and (8) with that of (19) and (20). Suppose $\|\mathbf{W}_n\| = \xi$, with $\xi > 0$ being a small constant, we have

$$\|(\mathbf{W}_n^T \mathbf{W}_n)^{-1} \mathbf{W}_n^T\| \|\mathbf{W}_n\| \quad (26)$$

$$\geq \|(\mathbf{W}_n^T \mathbf{W}_n)^{-1} \mathbf{W}_n^T \mathbf{W}_n\| = \|\mathbf{I}\| = 1, \quad (27)$$

where \mathbf{I} is an identity matrix. Hence, we obtain

$$\|(\mathbf{W}_n^T \mathbf{W}_n)^{-1} \mathbf{W}_n^T\| \geq \|\mathbf{W}_n\|^{-1} = \xi^{-1}. \quad (28)$$

On the other hand,

$$\|\mathbf{H}_{n+1}\| = \|(\mathbf{W}_n^T \mathbf{W}_n)^{-1} (\mathbf{W}_n^T \mathbf{V})\|. \quad (29)$$

As a consequence, $\|\mathbf{H}_{n+1}\|$ can grow and become unstable within one step. The hybrid algorithm instead remains stable under a given bound, since

$$\|\mathbf{H}_{n+1}\| \leq \|\mathbf{H}_n\| \left\| \frac{(\mathbf{W}_n^T \mathbf{V})}{(\mathbf{W}_n^T \mathbf{W}_n \mathbf{H}_n)} \right\| \approx \frac{\xi \|\mathbf{V}\|}{\xi^2} = \xi^{-1}. \quad (30)$$

We note that the hybrid algorithm inherits the stability of the LS algorithm while converging faster. In many of our tests in MATLAB, warnings for the condition number of $(\mathbf{W}_n^T \mathbf{W}_n)^{-1}$ from (7) have been observed, while our hybrid algorithm performs extremely well without such an issue.

4. SIMULATIONS

In the first experiment, we generated \mathbf{V} synthetically as the absolute value of a zero-mean Gaussian distributed random variable, and initialized \mathbf{W} and \mathbf{H} in the same way. The dimensions of these matrices were set as $m = 500, p = 400$, and $r = 4$. We performed 20 independent random tests in which both \mathbf{W} and \mathbf{H} were kept the same for all the three algorithms. The evolution of the cost function averaged over the 20 tests is shown in Figure 3 for the proposed algorithm, as well as the LS and ALS algorithms. From this figure, we observe that the proposed algorithm converges considerably faster than both the LS and ALS algorithm. This argument is especially true when the decomposition rank is relatively low as compared with the data dimension. In our experiments, this observation still holds for r being increased up to 30. When increasing r to 50, the ALS algorithm becomes unstable, while the proposed algorithm still converges even though its rate becomes lower than that of the LS algorithm. In the second experiment, we generated \mathbf{V} as the spectrogram

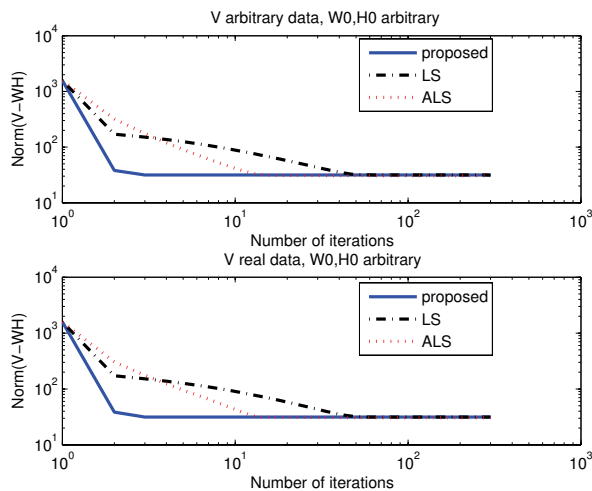


Fig. 3. Comparison of the average convergence behavior of the proposed algorithm, the LS algorithm and the ALS algorithm for the synthetically generated data.

of an audio signal C10_whistle.wav¹, where the frame length of the FFT was set to 512. Hence, the dimension of \mathbf{V} was $m = 512, p = 174$. Both \mathbf{W} and \mathbf{H} were randomly initialized. The rank r was set to 4, and 50% overlap between the windows was used for generating the spectrogram. All the three algorithms were applied to decompose the music notes from the audio signal. Their convergence curves averaged over 20 independent tests (with \mathbf{W} and \mathbf{H} randomly initialized) are shown in Figure 4, where the improved convergence performance of the hybrid algorithm can be clearly observed. The hybrid algorithm is currently under further investigation and being applied to a wider range of data, and more results will be reported shortly.

¹Available at www.ee.surrey.ac.uk/Personal/W.Wang/demondata.html

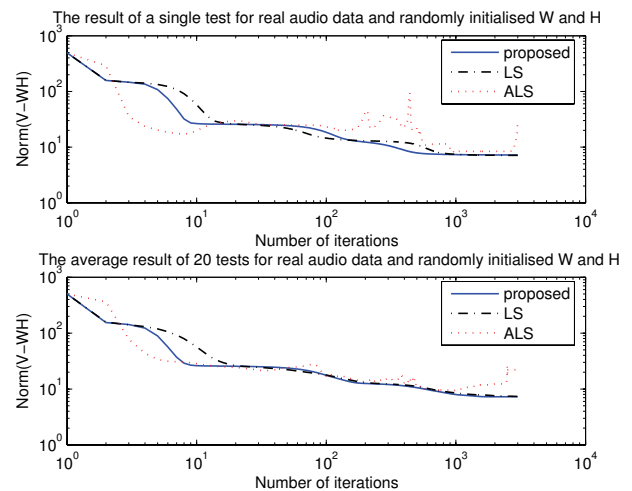


Fig. 4. Comparison of the average convergence behavior of the proposed algorithm, the LS algorithm and the ALS algorithm for the real audio data.

5. CONCLUSION

We have introduced a hybrid algorithm for NMF using the mixed iterations based on alternating least squares and Lee-Seung algorithms. An advantage of the hybrid framework lies in that it can be readily extended to encompass other possible iterations for enhancing their convergence performance. Both theoretical and numerical results have shown the advantage of the proposed hybrid method.

6. REFERENCES

- [1] R. Albright, J. Cox, D. Duling, A. Langville, and C. Meyer, "Algorithms, initializations, and convergence for the nonnegative matrix factorization," *NCSU Technical Report Math 81706*, 2006.
- [2] A. Ben Hamza and D. J. Brady, "Reconstructin of reflectance spectra using robust NMF," *IEEE Trans. on Signal Proc.* vol. 54, no. 9, 2006.
- [3] M. Berry, M. Browne, A. Langville, P. Pauca and R.J. Plemmons, "Algorithms and appliactions for approximation nonnegative matrix factorization," *Computational Statistics and Data Analysis*, 2006.
- [4] D. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts," in *Advances in Neural Inf. Process. Sys. (NIPS)*, vol. 17.
- [5] D. D. Lee and H. S. Seung, "Algorithms for Non-negative matrix factorization," *Avd. Neural Inf. Proc. Syst.* vol. 13, pp. 556-562, 2005.
- [6] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature* vol. 401, pp. 788-791, 1999.
- [7] D. Kim, S. Sra, and I. S. Dhillon, "Fast Newton-type methods for the least squares nonnegative matrix approximation Problem," in *Proc. of the 6th SIAM Int. Conf. on Data Mining*, pp. 343-354, April 2007.
- [8] C.J. Lin, "On the convergence of multiplicative update algorithms for nonnegative matrix factorization," *IEEE Trans. on Neural Networks*, vol. 18, pp. 1589-1596, 2007.
- [9] R. Zdenuk and A. Cichocki, "Nonnegative matrix factorization with quadratic programming," *Neurocomputing*, vol. 71, pp. 2309-2320, 2007.