Accelerating the Lee-Seung Algorithm for Nonnegative Matrix Factorization

Edward F. Gonzalez and Yin Zhang
Department of Computational and Applied Mathematics
Rice University, Houston, Texas 77005

March 3, 2005

Abstract

Approximate nonnegative matrix factorization is an emerging technique with a wide spectrum of potential applications in data analysis. Currently, the most-used algorithms for this problem are those proposed by Lee and Seung [7]. In this paper we present a variation of one of the Lee-Seung algorithms with a notably improved performance. We also show that algorithms of this type do not necessarily converge to local minima.

1 Introduction

Approximate nonnegative matrix factorizations are relatively new but promising techniques for data analysis, with possible utilities in dimension reduction, compression, feature extraction, pattern recognition, object detection and classification, to name a few amongst many potential applications. Because of contributions from widespread fields, literature on this subject is already quite large, and motivations and terminologies used are diverse. Roughly speaking, in terms of linear algebra, one tries to approximate a given data array (or training set) A of nonnegative values by a product of two nonnegative matrices, i.e., $A \approx WH$, where the number of columns of W may be much smaller than that of A. Those columns of W would presumably represent

the principal components, hidden concepts, prominent features, latent variables, ..., of the underlying data, depending on the application context.

The singular value decomposition (SVD) is a widely used factorization when no constraints are imposed on the factors W and H. In many applications, however, physical data in A can only take non–negative values, such as pixel values in imagery data. In such cases it may be more desirable to obtain physically meaningful "non-negative principal parts", and then represent data as additive combinations of these "parts". This naturally leads to the requirement that both W and H be nonnegative, hence the term *Approximate nonnegative matrix factorization* (ANMF). A solution to the ANMF problem can be obtained by solving the following optimization problem

$$\begin{array}{ll} \min & f(W,H) := \frac{1}{2}\|A - WH\|_F^2 \\ \text{s.t.} & W \geq 0,\ H \geq 0 \end{array} \tag{1}$$

where $A \in \Re^{m\times n}$, $W \in \Re^{m\times p}$, $H \in \Re^{p\times n}$, and $\|\cdot\|_F$ is the Frobenius norm. We note that other objective functions can be used to measure the error of the approximation instead of the Frobenius norm (which is the most appropriate only when errors are normally distributed). In practice, the number p is typically chosen to be significantly smaller then n; yet how to select p is still a difficult and important issue.

ANMF has found applications in the areas of face detection [8], speech recognition [10], text and video/audio document processing [5, 4] genetics [3], and even entertainment [11]. ANMF seems to be particularly promising in face detection [6], as a number of studies have found that the part-based additive nature of ANMF makes it less sensitive to occlusions and other unfavorable conditions.

In section 2, we state the popular Lee-Seung algorithm commonly used for solving the (ANMF) problem (1). We then give an accelerated version of the Lee-Seung algorithm in section 3. In section 4, we present numerical results to demonstrate that the accelerated algorithm indeed offers a better performance than that of the Lee-Seung algorithm. In section 5, we examine the question of whether these algorithms actually converge to the stationary points of the ANMF problem (1). Concluding remarks are given in the final section.

2 The Lee-Seung Algorithm

The popularity of the Lee-Seung algorithm for solving the ANMF problem (1) can be attributed to both its simplicity, which is evident in the fact that the algorithm can be carried out in just a few lines of Matlab code, and its practical performance. As we will see, the Lee-Seung algorithm can be viewed as a diagonally-scaled gradient-type algorithm. It requires a relatively low computational cost on a per iteration basis, and it has been reliable in generating solutions of adequate quality.

To motivate the algorithm, we observe that for a fixed W (or H), problem (1) becomes a linear least squares problem in terms of H (or W). The Lee-Seung algorithm successively updates H and W, which fixing the other, by taking a step in a certain weighted negative gradient direction for the function $f(W, H)$ as defined in (1); namely,

$$H_{ij} \leftarrow H_{ij} - \eta_{ij} \left[\frac{\partial f}{\partial H}\right]_{ij} \equiv H_{ij} + \eta_{ij} \left(W^T A - W^T W H\right)_{ij}, \tag{2}$$

$$W_{ij} \leftarrow W_{ij} - \zeta_{ij} \left[\frac{\partial f}{\partial W}\right]_{ij} \equiv W_{ij} + \zeta_{ij} \left(A H^T - W H H^T\right)_{ij}, \tag{3}$$

where η_{ij} and ζ_{ij} are individual weights for the corresponding gradient elements. Lee and Seung [7] use the following weights,

$$\eta_{ij} = \frac{H_{ij}}{(W^T W H)_{ij}}, \quad \zeta_{ij} = \frac{W_{ij}}{(W H H^T)_{ij}}, \tag{4}$$

and arrive at the updating formulas:

$$H_{ij} \leftarrow H_{ij} \frac{(W^T A)_{ij}}{(W^T W H)_{ij}}, \quad W_{ij} \leftarrow W_{ij} \frac{(A H^T)_{ij}}{(W H H^T)_{ij}}. \tag{5}$$

Once started from a positive pair (W, H), these updates will always maintain positivity in the subsequent iterates. In addition, Lee and Seung [7] have also proved that the algorithm, which we will refer to as the (LS) algorithm, will monotonically decrease the objective function $f(W, H)$; i.e., for any two successive iterations k and $k+1$,

$$\|A - W^{k+1} H^{k+1}\|_F \leq \|A - W^k H^{k+1}\|_F \leq \|A - W^k H^k\|_F, \tag{6}$$

where strictly inequalities hold unless the gradient of $f(W, H)$ vanishes. However, it needs to be pointed out that the monotone decrease in objective value by no

means guarantees convergence to a local minimum of the ANMF problem (1), as was incorrectly claimed in [7]. In fact, we will present numerical evidence in Section 5 showing that in general the (LS) algorithm does not appear to converge to local minima of the ANMF problem.

3 Accelerating the (LS) Algorithm

We observe that the optimal H relative to a fixed W can be obtained, column by column, by solving separately

$$\begin{array}{ll} \min & \frac{1}{2}\|Ae_j - WHe_j\|_2^2 \\ \text{s.t.} & He_i \geq 0, \end{array} \tag{7}$$

where e_j is the j^{th} column of the $n \times n$ identity matrix. Similarly, we can obtain the optimal W relative to a fixed H by solving, row by row,

$$\begin{array}{ll} \min & \frac{1}{2}\|A^T e_i - H^T W^T e_i\|_2^2 \\ \text{s.t.} & W^T e_i \geq 0, \end{array} \tag{8}$$

where e_i is the i^{th} column of the $m \times m$ identity matrix. The above problems can all be cast into a common form of the convex quadratic program

$$\begin{array}{ll} \min & \frac{1}{2}\|Ax - b\|_2^2 \\ \text{s.t.} & x \geq 0, \end{array} \tag{9}$$

where $A \geq 0$ and $b \geq 0$. In this problem, the variables as well as the given data are all nonnegative. It is therefore called a totally nonnegative least squares (TNNLS) problem.

The (LS) algorithm can be viewed as successively improving the two sets of TNNLS problems, (7) and (8), using an update rule $x \leftarrow x + p$, where p is the negative of a certain weighted gradient of the corresponding objective function for each TNNLS problem in (7) and (8) (see (2), (3) and (4)).

In [9], Merritt and Zhang proposed an interior–point gradient method for solving the TNNLS problems in which they use a search direction p equivalent to that used in the (LS) algorithm. However, they use the update rule: $x \leftarrow x + \alpha p$ with a step-length α, which is taken to be the exact minimizer of the objective function in the search

direction p if this creates a positive update. Otherwise, α is chosen to be τ, $\tau \in (0,1)$, times the longest step-length possible in the p direction that ensures $x + \alpha p \geq 0$. In either case, the algorithm always keeps the iterates positive (hence an interior-point algorithm) and always decrease the objective function. Merritt and Zhang [9] have proved that for τ sufficient close to 1, the inequality $\alpha > 1$ will hold. Moreover, they have also established the convergence of their algorithm.

We propose to modify the (LS) algorithm by applying the same update rule with step-length α used in [9] to the successive updates in improving the objective functions in the two sets of TNNLS problems in (7) and (8). This results in a modified version of the Lee-Seung algorithm that successively updates the matrix H column by column and W row by row, with an individual step-length α for each column of H and an individual step-length β for each row of W. The algorithm can be written as the following update rule: starting from some positive pair (W, H), do

$$H_{ij} \leftarrow H_{ij} + \alpha_j \, \eta_{ij} \left(W^T A - W^T W H \right)_{ij}, \tag{10}$$

$$W_{ij} \leftarrow W_{ij} + \beta_i \, \zeta_{ij} \left(A H^T - W H H^T \right)_{ij}, \tag{11}$$

where η_{ij} and ζ_{ij} are defined as in (4), and the step-length parameters α_j, $j = 1, 2, \cdots, n$, and β_i, $i = 1, 2, \cdots, m$, are computed as follows. Instead of enumerating all the problems in (7) and (8), it suffices to consider the common form of the TNNLS problem (9). Let $x > 0$,

$$q = A^T(b - Ax) \quad \text{and} \quad p = [x \,./\, (A^T A x)] \circ q,$$

where the symbols "./" and "$\circ$" denote component-wise division and multiplication, respectively. Then we define for some $\tau \in (0,1)$

$$\alpha = \min \left(\frac{p^T q}{p^T A^T A p}, \; \tau \, \max\{\hat{\alpha} : x + \hat{\alpha} p \geq 0\} \right). \tag{12}$$

By replacing (A, b, x) by (W, Ae_j, He_j) (or $(H^T, A^T e_i, W^T e_i)$), respectively, we would immediately obtain the corresponding step-length formula for α_j (or β_i). As can be seen, q is the negative gradient of the objective function in (9), and the search direction p is a diagonally scaled negative gradient direction. The step length α is either the minimum of the objective function in the search direction, or a τ-fraction of the step to the boundary of the nonnegative orthant.

It has been shown in [9] that both quantities, $p^T q / p^T A^T A p$ and $\max\{\hat{\alpha} : x + \hat{\alpha} p \geq 0\}$, in the definition of the step α are greater than 1. Hence we can make $\alpha_j \geq 1$ and $\beta_i \geq 1$ by choosing τ sufficiently close to one. In our implementation, we set $\tau = 0.99$ which practically ensures that the step length parameters are almost always greater than one.

It has also been shown in [9] that in their interior-point algorithm the objective function is monotonically decreasing. Hence in each of the individual problems in (7) and (8) the objective value must be monotonically decreasing. Consequently, when given H and W are updated through the formulas (10) and (11), respectively, the monotone-decreasing property of (6) is still satisfied, as is in the (LS) algorithm.

Obviously, when the step-length parameters are set to one, update formulas (10) and (11) reduce to the Lee-Seung updates (2) and (3). In our modified algorithm, the step-length parameters are allowed to be greater than one. This implies that at any given (W, H), we can achieve at least the same, and likely greater, amount of decrease in the objective function than the Lee-Seung algorithm can. For this reason, we will call the proposed algorithm the accelerated Lee-Seung (ALS) algorithm. In the next section, we will demonstrate in our numerical experiments that the (ALS) algorithm indeed produces better results than the (LS) algorithm, especially when only a limited number of iterations are performed.

4 Numerical Experiments

In this section, we empirically compare the performance of the accelerated Lee-Seung algorithm (ALS) with that of the original Lee-Seung algorithm (LS). These algorithms were implemented in Matlab and applied to the CBC face database [2] and to the ORL face database [1]. These databases have been frequently used in the literature to evaluate algorithm performance.

We create data matrices A's from the above databases so that each column represents a particular picture. The size of the matrices produced from the CBC and ORL databases are (361×2429) and (10304×400), respectively, as the CBC database contains 2429 pictures of (19×19) pixel resolution, while the ORL database contains 400 pictures of (112×92) pixel resolution. For the sake of computing time, we also used a reduced ORL database, R-ORL, in which we reduce the pixel resolution from

the original (112×92) to a (56×46) resolution, thus resulting a data matrix of size (2576×400). For further details on these databases the reader is referred to the references [2, 1].

In our experiments, we apply the the (LS) and (ALS) algorithms to the ANMF problem (1) to compute approximate solutions with various values of p. In order to get a fair comparison, we followed the steps below for each test case.

1. Always use the same randomly generated starting point for both algorithms.
2. Run the (LS) algorithm for a given number of iterations, and record the CPU time used.
3. Then run the (ALS) algorithm, and stop it once the CPU time used is equal to or greater than that used by the (LS) algorithm.

We note that an iteration of the (ALS) algorithm takes more time than that of the (LS) algorithm, as the former needs additional time to compute the step-length parameters at each iteration. Steps 2 and 3 in our experiments ensure that a fair comparison is carried out so that both algorithms are run for approximately the same amount of time for each test case.

To compare the performance of the two algorithms in terms of solution quality, we define the relative improvement of the (ALS) algorithm over the (LS) algorithm as the quantity:

$$\text{Relative Improvement} = \frac{f_{LS} - f_{ALS}}{f_{LS}} \times 100\%$$

where f_{LS} is the objective value $f(W, H)$ evaluated at the final iterate pair returned by the (LS) algorithm after a given number of iterations; and similarly f_{ALS} is the objective value evaluated at a pair returned after the given amount of run time, as dictated by the number of iterations taken by the (LS) algorithm. The relative improvement would be negative if the (ALS) algorithm does not offer an improvement in the objective value over the (LS) algorithm.

Table 1 shows the relative improvement of (ALS) over (LS) for four runs on each of the three data sets (CBC, the reduced and unreduced ORL) corresponding to four different p values. In each test case, we ran the (LS) algorithm with five different iteration numbers, as given in the table.

Table 1: The relative improvement of (ALS) over (LS) is give in percentage. The (LS) algorithm was run a given number of iterations, then the (ALS) algorithm was run for an equivalent amount of CPU time. The number of iterations used by (ALS) is given in parenthesis.

		Number of iterations for (LS)				
Database	p	20	50	100	200	400
CBC	25	27.1% (13)	10.3% (31)	5.4% (62)	3.1% (145)	1.4% (221)
CBC	36	29.4% (13)	16.4% (32)	8.4% (63)	5.9% (142)	1.6% (228)
CBC	100	38.2% (14)	25.7% (33)	16.6% (65)	13.5% (148)	8.4% (259)
CBC	121	37.5% (14)	26.4% (33)	18.0% (65)	15.3% (146)	8.5% (253)
R-ORL	25	40.8% (14)	32.8% (34)	16.3% (66)	5.9% (131)	2.6% (261)
R-ORL	36	43.9% (14)	29.8% (33)	17.4% (65)	7.2% (130)	2.4% (257)
R-ORL	100	48.5% (13)	38.7% (33)	27.3% (64)	15.9% (127)	7.7% (254)
R-ORL	121	49.6% (13)	41.1% (32)	30.0% (64)	18.5% (128)	9.5% (255)
ORL	25	39.5% (13)	28.9% (32)	14.3% (62)	5.3% (124)	2.9% (249)
ORL	36	43.9% (13)	31.1% (31)	18.9% (62)	8.6% (123)	3.8% (245)
ORL	100	49.8% (14)	40.7% (33)	29.8% (65)	17.8% (129)	8.4% (258)
ORL	121	50.4% (14)	41.8% (33)	32.6% (65)	20.2% (131)	9.9% (260)

We observe from the test results that (i) the values of the relative improvement are always positive, and (ii) for each database the relative improvement increases as the p-value increases; (iii) for any given p value, the larger the database size is, the lager is the relative improvement; and (iv) for a given p value, the relative improvement becomes smaller when the algorithms were run a longer time (thus allowing the slower (LS) algorithm to recover some lost ground). The largest test case is when 400 iterations were given to the (LS) algorithm to run on the ORL database with $p = 121$, which took around $3,200$ CPU seconds on a SunBlade 1000 workstation.

To see whether or not a relative improvement of (ALS) over (LS) on the objective value actually translates into a qualitative improvement, in Figure 1 we show four randomly chosen pictures from the (unreduced) ORL dataset and their approximations by low-rank nonnegative matrix factorization $A \approx WH$. The original pictures, the (LS) approximations and the (ALS) approximations are listed in columns 1, 2

and 3, respectively. The approximations were generated for $p = 100$ with 100 iterations given to the (LS) algorithm and the same amount of CPU time to the (ALS) algorithm (which was over 650 CPU seconds on a SunBlade 1000 workstation). In this case, the relative improvement of (ALS) over (LS) is about 30%.

Figure 1: Four pictures (in column 1) from the ORL dataset, and the corresponding (LS) and (ALS) approximations (in columns 2 and 3, respectively).

As can been seen from Figure 1, the differences in the quality of approximations by the two algorithms are clearly visible for all the four pictures.

Our experiment results clearly show that the the accelerated Lee-Seung algorithm indeed converges notably faster, especially at early stages. This property should be particularly useful for large-scale or real-time applications.

5 Do the algorithms converge to local minima?

The Karush-Kuhn-Tucker (KKT) condition for the ANMF problem (1) can be written into the following two equations:

$$\begin{aligned} \min(H,\, W^T W H - W^T A) &= 0, \\ \min(W,\, W H H^T - A H^T) &= 0, \end{aligned} \tag{13}$$

where the minimum is taken component–wise. The first expression, for example, states that both H and $W^T W H - W^T A \equiv \partial f / \partial H$ should be component-wise non-negative and at least one (the smaller of the two) should be zero. Hence the equations imply both feasibility and complementarity in a compact form.

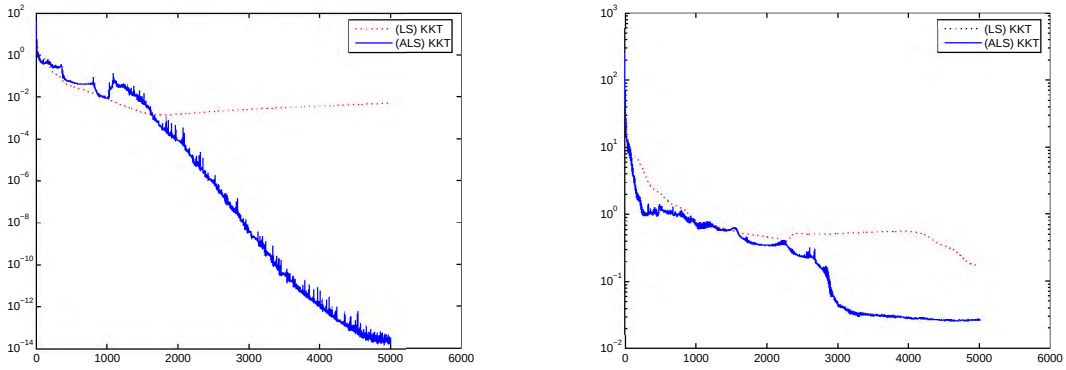
By well-known optimization theory, the KKT conditions (13) are necessary for every local minimum of the ANMF problem (1). In particular, if a sequence converges to a local minimum of (1), then the residual norm of the KKT equations (i.e., the left-hand-side of (13)) must go to zero.

We have conducted numerical experiments to examine the issue whether or not the (LS) and (ALS) algorithms converge to local minima of (1). In our experiments, we used randomly generated data matrices of various sizes, and allowed the two algorithms to run a large number of iterations, while monitoring the size of the KKT residual measured by the L_1 vector norm.

Figure (2) contains two illustrative cases from our experiments in which the KKT residual norm is plotted for 5000 iterations on two small date matrices. In the pictures, the sizes of the data matrices A are given, and so is the value of p that is the number of columns in W (or rows in H).

The results shown in figure (2) are typical of both the (LS) and (ALS) algorithms. We consider that convergence has occurred if the KKT residual norm becomes close to machine precision. Our experiments have shown that (i) on very small problems, the (ALS) algorithm tends to converge to KKT points more consistently than the (LS) algorithm; in fact we have not observed a single case where the former did not converge while the latter did; (ii) on larger problems, both algorithms do not seem to converge to KKT points within a large number of iterations.

Based on our experiments, we feel safe to infer that for problems of practically meaningful sizes, one should not expect that either algorithm converges to local minima, even though they may produce otherwise satisfactory results. From an opti-



(a) A is 10×7, $p = 4$

(b) A is 25×15, $p = 10$

Figure 2: Behavior of KKT residual norm for two runs

mization point of view, this lack of reliable convergence has to be considered rather unsatisfactory.

6 Concluding Remarks

By introducing well defined step-length parameters, we derive an accelerated version of the Lee and Seung algorithm for the approximate nonnegative matrix factorization problem in Frobenius norm. Our extensive numerical experiments have shown that the accelerated algorithm consistently outperforms the Lee-Seung algorithm by a notable margin, especially when computing time is limited. The faster convergence of the accelerated algorithm is desirable for large-scale or real-time applications.

In principle, the same idea could be applied to the other Lee-Seung algorithm based on minimizing the divergence function (see [7]). However, since a close-form step-length formula does not seem possible in that case, a line search would be necessary to find a good step length, making the modification less appealing.

We have also demonstrated that, contrary to the claim of guaranteed convergence, the Lee-Seung algorithm, and our accelerated algorithm as well, do not appear to converge to local minima within a realistic amount of run time for problems of meaningful sizes. This behavior is certainly undesirable from an optimization point

of view. It remains a research issue to develop theoretically sound and practically efficient algorithms for the approximate nonnegative matrix factorization problem.

References

[1] AT&T laboratories, Cambridge ORL database of faces.
Available at: *http://www.uk.research.att.com/facedatabase.html*.

[2] M.I.T. Center for Biological and Computaional Learning CBCL face database.
Available at: *http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html*.

[3] J. Brunet, T. Golub, and J. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. In *Proceedings of the National Academy of Sciences*, volume 101, pages 4164–4169, 2004.

[4] Matthew Cooper and Jonathan Foote. Summarizing video using non-negative similarity matrix factorization, December 2002. IEEE Multimedia Signal Processing Workshop.

[5] C.W. Lee, K. Jung, H. Kang, and H. J. Kin. Font classification using NMF, August 2003. Lecture Notes in Computer Science (LNCS 2756).

[6] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, October 1999.

[7] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[8] Stan Z. Li, Xin Wen Hou, Hong Jiang Zhang, and Qian Sheng Cheng. Learning spatially localized parts-based representation. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, volume I, pages 207–212, 2001.

[9] Michael Merritt and Yin Zhang. An interior-point gradient method for large-scale totally nonnegative least squares problems. In *J. Optimization Theory and Applications*, volume 126, No.1, pages 191-202, July 2005.

[10] Miroslav Novak and Richard Mammone. Improvement of non-negative matrix factorization based language model using exponential models. In *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 190–193, 2001.

[11] Frank Wang, Alan Edelman, and Arun Rao. An application of NNMF in a color calibration algorithm of film. Technical report, MIT Laboratory for Computer Science, 2003.