Block-Iterative Algorithms for Non-Negative Matrix Approximation

Suvrit Sra
Max-Planck Institut für biologische Kybernetik
72076 Tübingen, Germany
suvrit.sra@tuebingen.mpg.de

Abstract

In this paper we present new algorithms for non-negative matrix approximation (NMA), commonly known as the NMF problem. Our methods improve upon the well-known methods of Lee & Seung [12] for both the Frobenius norm as well the Kullback-Leibler divergence versions of the problem. For the latter problem, our results are especially interesting because it seems to have witnessed much lesser algorithmic progress as compared to the Frobenius norm NMA problem. Our algorithms are based on a particular ***block-iterative*** *acceleration technique for EM, which preserves the multiplicative nature of the updates and also ensures monotonicity. Furthermore, our algorithms also naturally apply to the Bregman-divergence NMA algorithms of [6]. Experimentally, we show that our algorithms outperform the traditional Lee/Seung approach most of the time.*

1. Introduction

Non-negative matrix approximation (NMA), which was introduced under the name *positive matrix factorization* by [15], and later popularized as *non-negative matrix factorization* or NMF by [12], is a popular matrix decomposition technique that is now increasingly employed for data analysis. In its most common incarnation, given a non-negative input matrix A, NMA seeks to minimize

$$\mathscr{F}(\mathsf{A};\hat{\mathsf{A}}) = \|\mathsf{A}-\hat{\mathsf{A}}\|_F^2, \tag{1.1}$$

where $\hat{\mathsf{A}} = \mathsf{WH}$ and $\mathsf{W},\mathsf{H} \geq 0$. Several algorithms exist for this Frobenius norm version of NMF and its variations—see for e.g., [12, 11, 2, 20]. The Kullback-Leibler (KL) divergence NMA problem that seeks to minimize

$$\mathscr{F}(\mathsf{A};\hat{\mathsf{A}}) = \sum\nolimits_{ij} a_{ij}\log(a_{ij}/\hat{a}_{ij}) - a_{ij} + \hat{a}_{ij}, \tag{1.2}$$

where $\hat{\mathsf{A}} = \mathsf{WH}$ and $\mathsf{W},\mathsf{H} \geq 0$, has received much lesser attention. Both (1.1) and (1.2) are special cases of the more general Bregman-divergence NMA problem [6]

$$\mathscr{F}(\mathsf{A};\hat{\mathsf{A}}) = \sum\nolimits_{ij} D_\phi(a_{ij};\hat{a}_{ij}), \tag{1.3}$$

where $\hat{\mathsf{A}}$ is as usual and we define the *Bregman-divergence*

$$D_\phi(x;y) = \phi(x) - \phi(y) - \phi'(y)(x-y), \tag{1.4}$$

for a given strictly convex function ϕ[1]. With $\phi(x) = \frac{1}{2}x^2$ one obtains (1.1) from (1.3), and with $\phi(x) = x\log x$ one obtains (1.2). Note that since D_ϕ is asymmetric we also consider the NMA problem that seeks to minimize

$$\mathscr{F}(\hat{\mathsf{A}};\mathsf{A}) = \sum\nolimits_{ij} D_\phi(\hat{a}_{ij};a_{ij}). \tag{1.5}$$

We derive new algorithms for (1.1)–(1.3), and (1.5). Our approach highlights important connections to previously known work and sheds light on the genesis of the well-known Lee & Seung algorithms. In fact, we follow in the footsteps of Lee&Seung [12], who extended the vector based EM methods to NMA via alternating descent, and obtain block-iterative algorithms for NMA by building on techniques used for accelerating the EM algorithm.

1.1. Related Work

Exact NMF, i.e., where $\mathsf{A} = \mathsf{WH}$, was studied in linear algebra in the 70s [14]. Later Paatero et al. studied NMA under the name of positive matrix factorization [15]. However, NMA became popular after Lee&Seung described simple alternating descent algorithms for it in [12]. Since then interest in this problem has literally exploded; we refer the reader to the articles [2, 20, 13] for extensive references.

Most of the algorithmic progress on NMA has concentrated on (1.1) with a few exceptions [6, 9, 4]. In this paper we develop new algorithms that apply to not only (1.1) but also to (1.2), (1.3), and (1.5). Subject to certain convexity assumptions, our techniques also extend to other loss functions. However, we limit our discussion to Bregman divergences because of lack of space.

Algorithmically, the most popular approach to NMA is alternating descent [12, 6], though some researchers have also considered alternating minimization [11, 2] for (1.1). If

[1] Actually ϕ must satisfy certain other technical conditions [3, 1], but for our purposes strict convexity is enough.

we trace the origins of the well-known Lee/Seung NMA algorithms, we see that they are essentially generalizations (to matrices via alternating descent) of the successful EMML algorithm of [19], or the ISRA algorithm of [5]. This connection has been largely ignored in the literature (except by some authors [4]), and it turns out to be particularly important because the EMML algorithm and its derivatives have been the subject of intense study in the field of medical imaging [17]—Indeed, one can exploit some of the developments for EMML in a manner analogous to Lee/Seung [12]. In fact we go even further, because we not only handle more general objective functions but also permit principled incorporation of regularization terms.

2. Background

To avoid clutter, we concentrate on the two factor version of NMA where $\hat{\mathsf{A}} = \mathsf{WH}$, i.e., we aim to compute $\mathsf{W}, \mathsf{H} \geq 0$ such that $\mathscr{F}(\mathsf{A}; \hat{\mathsf{A}}) = \mathscr{F}(\mathsf{A}; \mathsf{WH})$ is minimized. Owing to the product BC the resulting problem is almost never convex, whereby it is unrealistic to expect to find a globally optimal solution; this intuition is supported by the recent formal proof of the NP-Hardness of exact NMA [21]. Many times, for instance in (1.1), (1.2), and (1.5), the objective function is individually convex in W and H, which allows one to invoke an alternating minimization procedure.

However, two main difficulties arise. First, even performing alternating minimization can be too expensive, especially when the matrices become large. Second, it could easily happen that the objective function is *not even* individually convex in W or H, for e.g., often for (1.3). Therefore, just settling for alternating descent, instead of minimization becomes a practical choice. The general algorithmic scheme is summarized in Figure 1.

1. Initialize W and/or H; set $t \leftarrow 0$.
2. Fix W; find H' s.t. $\mathscr{F}(\mathsf{W}, \mathsf{H}') \leq \mathscr{F}(\mathsf{W}, \mathsf{H})$,
3. Fix H'; find W' s.t. $\mathscr{F}(\mathsf{W}', \mathsf{H}') \leq \mathscr{F}(\mathsf{W}, \mathsf{H}')$,
4. Repeat steps 2 & 3 to convergence

Figure 1. Alternating Descent Scheme

Our algorithms follow this general scheme, as do the Lee & Seung algorithms and most other NMA methods. Assuming that our distortion function $\mathscr{F}$ is separable, i.e.,

$$\mathscr{F}(\mathsf{A}; \mathsf{WH}) = \sum\nolimits_j F(\mathsf{a}_j; \mathsf{Wh}_j),$$

where F denotes the column-wise distortion, we can just focus on solving the core problem

$$\min_{\mathsf{h} \geq 0} \; F(\mathsf{a}; \mathsf{Wh}), \tag{2.1}$$

for an arbitrary column h of matrix H, and corresponding column a of A (similarly for rows). If we solve (2.1) "exactly" over all columns h, then the resulting solution H^{t+1} clearly satisfies Step 3 of the above scheme. For the Frobenius norm NMA problem, such an approach is followed by [11], who derived an alternating non-negatively constrained least-squares algorithm. In general, we will be satisfied with "inexact" solutions to (2.1), where we merely ensure descent in F (and thereby in $\mathscr{F}$).

2.1. Auxiliary functions

A simple approach to obtaining descent algorithms for (2.1) is to first find an *auxiliary function* $G(\mathsf{h}; \tilde{\mathsf{h}})$ that satisfies

1. $G(\mathsf{h}; \mathsf{h}) = F(\mathsf{a}; \mathsf{Wh})$ for all $\mathsf{h} \geq 0$
2. $G(\mathsf{h}; \tilde{\mathsf{h}}) \geq F(\mathsf{a}; \mathsf{Wh})$ for all $\mathsf{h}, \tilde{\mathsf{h}} \geq 0$.

The function G is chosen so that it *decouples* the variables h that occur as a linear combination Wh. If F is a convex function of h, then it is easy to construct (see [6, 16] for details) an auxiliary function G that is easier to optimize. For example, the famous EM algorithm is essentially built around exploiting the convexity of the $-\log x$ function for obtaining an auxiliary function in the E-Step. We remark that one need not always explicitly construct such auxiliary functions. For example, in [6] the authors derive procedures for (1.3) that are not based on auxiliary functions.

2.2. Block-iterative algorithms

We now present some background about some block-iterative algorithms, especially with the motive of obtaining new algorithms for NMA. Experience with EM has shown that methods based on auxiliary functions usually converge rather slowly. To overcome this hurdle a vast amount of effort has been invested in trying to speed up EM. One of the simplest (and most successful) ideas has been to divide W into blocks, and to build an auxiliary function for each block. With this procedure one can still make progress over each block without increasing the overall computational costs. For minimizing the KL-divergence (for vectors) such a *block-iterative* method was popularized under the name ordered subsets EM (OSEM) by [10]. In this paper we extend the block-iterative idea in three main directions:

1. Block-iteration for NMA via alternating descent for matrices (§§ 3, 4),
2. Extension to general objective functions such as Bregman-divergences,
3. Extension to NMA problems *without* using the concept of auxiliary functions (Section 4).

3. Algorithms based on auxiliary functions

Formally, without loss of generality let the rows (indices) of W be partitioned into p disjoint blocks (or subsets)

$S_1, \ldots, S_p$ such that $\cup_i S_i = \{1, \ldots, m\}$, where $\mathsf{a} \in \mathbb{R}^m_+$. We process the data by going through these subsets one by one, indexing the current subset of interest by s. Let t denote the index for one complete pass through all the subsets. Since we assumed separability of $\mathscr{F}$ over both rows and columns, we can write

$$F(\mathsf{a}; \mathsf{Wh}) = \sum_{s=1}^{p} \sum_{i \in S_s} F(a_i; [\mathsf{Wh}]_i).$$

Assuming F is convex, we can easily generate auxiliary functions for each block of rows S_s separately. For simplicity, we use the same form of auxiliary functions for each block, though this is not a necessity. Let $G_s(\mathsf{h}, \tilde{\mathsf{h}})$ denote the auxiliary function for block s, then we have the updates

$$\mathsf{h}^{(k,s+1)} = \operatorname*{argmin}_{\mathsf{h} \geq 0} G_s(\mathsf{h}, \mathsf{h}^{(k,s)}), \tag{3.1}$$

where $\mathsf{h}^{(1,1)} = \mathsf{h}^t$, and $\mathsf{h}^{t+1} = \mathsf{h}^{(k+\delta,p+1)}$, i.e., the value obtained after cycling through all the blocks δ times. It is easy to see that going through all the blocks essentially involves the same number of operations as going through all the rows at the same time (which is tantamount to using $p = 1$). The Lee & Seung-type NMA algorithms go through all the rows *just once* ($\delta = 1$) and return the resulting h^{t+1}. We use the index k in (3.1) to highlight the fact that we permit our algorithm to perform a few iterations ($\delta \in \{1, 4\}$) of (3.1) before obtaining h^{t+1}. In practice, this choice can lead to better objective function values.

We are now ready to look at important specific instances of the block-iterative scheme.

Example 1 (Block-KLNMA)**.** *The standard auxiliary function based update for the KLNMA problem (1.2) is [12]*

$$h_j^{t+1} = h_j^t \frac{1}{\sum_i w_{ij}} \sum_i \frac{a_i w_{ij}}{[\mathsf{Wh}^t]_i}.$$

Using an appropriate auxiliary function we can replace this with a block-iterative update, where for each block $s = 1, \ldots, p$, we have

$$h_j^{(k,s+1)} = h_j^{(k,s)} \frac{1}{\sum_{i \in S_s} w_{ij}} \sum_{i \in S_s} \frac{a_i w_{ij}}{[\mathsf{Wh}^{(k,s)}]_i}. \tag{3.2}$$

We set $\mathsf{h}^{(1,1)} = \mathsf{h}^t$, $\mathsf{h}^{(k+1,1)} = \mathsf{h}^{(k+1,p+1)}$, and $\mathsf{h}^{t+1} = \mathsf{h}^{(k+\delta,p+1)}$ where $\delta \in \{1, 4\}$. Similarly we can perform blocked updates for W by partitioning the columns of the matrix H into subsets. This alternation between W and H, as well as performing just a few iterations of (3.2) distinguish our approach from the vector version of the block-iterative updates of [10].

Lemma 1 (Monotonicity)**.** *To establish monotonicity of the algorithm based on* (3.2) *it suffices to show that*

$$\sum_{i \in S_s} \left(a_i - [\mathsf{Wh}^{(k,s+1)}]_i\right) \log \frac{[\mathsf{Wh}^{(k,s+1)}]_i}{[\mathsf{Wh}^{(k,s)}]_i} \geq 0. \tag{3.3}$$

Proof. Inequality (3.3) measures the decrease in the objective function restricted to subset S_s, and adding over all subsets we obtain the net decrease, hence overall monotonicity. This inequality follows easily by two applications of the log-sum inequality in combination with the update (3.2), and is omitted for brevity.

Example 2 (Block-LSNMA)**.** *The block-iterative version for the Frobenius norm NMA problem is*

$$h_j^{(k,s+1)} = h_j^{(k,s)} \frac{\sum_{i \in S_s} w_{ij} a_i}{\sum_{i \in S_s} w_{ij} [\mathsf{Wh}^{(k,s)}]_i}, \tag{3.4}$$

where we set $\mathsf{h}^{(1,1)}$ and h^{t+1} as in Example 1.

Monotonicity of (3.4) follows by an easy generalization of proof of unblocked version [12] combined with adding across all subsets.

Example 3 (Block-BregNMA)**.** *The block-iterative version of Bregman-divergence NMA algorithms of [6] is*

$$\phi'(h_j^{(k,s+1)}) = \phi'(h_j^{(k,s)}) \frac{\sum_{i \in S_s} w_{ij} \phi'(a_i)}{\sum_{i \in S_s} w_{ij} \phi'([\mathsf{Wh}^{(k,s)}]_i)}. \tag{3.5}$$

Note that these updates are of the form $\phi'(h_j^{t+1} = \phi'(h_j^t)\eta_j$, and one must compute the inverse of the function ϕ'. Fortunately, this can usually be done in closed form.

4. Non-auxiliary function methods

The authors of [6] presented a heuristic approach for minimizing (1.3), which was shown to empirically decrease the objective function monotonically. In our description of block-iterative algorithms above, the concept of auxiliary functions was used only for establishing monotonicity guarantees, not for applying the block-iterative scheme itself. Therefore, we easily obtain block-iterative versions of the algorithms of [6].

Example 4 (Block-BregNMA2)**.** *The block-iterative version of Bregman-divergence NMA algorithms of [6] corresponding to (1.3) is*

$$h_j^{(k,s+1)} = h_j^{(k,s)} \frac{\sum_{i \in S_s} w_{ij} \phi''([\mathsf{Wh}^{(k,s)}]_i) a_i}{\sum_{i \in S_s} w_{ij} \phi''([\mathsf{Wh}^{(k,s)}]_i) [\mathsf{Wh}^{(k,s)}]_i}, \tag{4.1}$$

where we set $\mathsf{h}^{(1,1)}$ and h^{t+1} as in Example 1.

4.1. Handling Regularization

Assume that we have the following regularized subproblem

$$\min_{\mathsf{h}\geq 0} \; F(\mathsf{a};\mathsf{Wh}) + \mu Q(\mathsf{h}), \tag{4.2}$$

where $\mu > 0$ is the regularization parameter. A particularly simple method to tackle (4.2) is the so-called one-step late (OSL) heuristic [8] that was rediscovered in the context of NMA by [6]. Here, while deriving updates such as (3.2)–(4.1) one replaces $\nabla[Q(\mathsf{h})]_j$ by $\nabla[Q(\mathsf{h}^t)]_j$ (i.e., the derivative at the previous step). For example, for (3.2) this leads to the update

$$h_j^{(k,s+1)} = \frac{h_j^{(k,s)}}{\sum_{i\in S_s} w_{ij} + \mu[\nabla Q(\mathsf{h}^{(k,s)})]_j} \sum_{i\in S_s} \frac{a_i w_{ij}}{[\mathsf{Wh}^{(k,s)}]_i}.$$

However, despite its simplicity this update can be invalid because it does not assure monotonicity like its unregularized counterpart (3.2). In general for Problem (4.2) if F is as in Section 3 and Q is the form $Q(\mathsf{h}) = \sum_j q([\mathsf{Ph}]_j)$, where q is also convex, then we can construct an auxiliary function $G + \bar{Q}$ as in §2.1 and use the update

$$\mathsf{h}^{(k,s+1)} = \operatorname*{argmin}_{\mathsf{h}\geq 0} G_s(\mathsf{h},\mathsf{h}^{(k,s)}) + \mu\bar{Q}(\mathsf{h};\mathsf{h}^{(k,s)}). \tag{4.3}$$

Further note that whenever (4.3) is not solvable in closed form, we can perform a few steps of Newton's method to obtain an approximate solution. The auxiliary functions G and $\bar{Q}$ are selected to decouple the variables h, whereby invoking Newton's method or some other non-linear equation solver is feasible for (4.3). However, Newton's method or other typical convex optimization methods could become impractical for the original problem (4.2) due to the non-negativity constraints and subproblem sizes.

5. Experiments

We now show experiments that demonstrate how block-iteration helps to improve upon the standard algorithmic techniques. The methods that we compare are:

1. LSNMA–Frobenius norm algorithm of [12] against BI-LSNMA, our block-iterative version based on (3.4)
2. KLNMA–KL-Divergence algorithm of [12] against BI-KLNMA, our block-iterative version based on (3.2).
3. BREGNMA of [6] against BI-BREGNMA, our block-iterative version based on (4.1)

LSNMA, KLNMA, and BREGNMA were obtained from *http://www.cs.utexas.edu/ suvrit/work/progs/nnma.m*. Our algorithms were also implemented in MATLAB. To ensure fairness for LSNMA and KLNMA, we edited the BREGNMA source-code to make it more efficient. Our experiments show that within the same running time, the block-iterative versions of the NMA codes obtain better objective function values than the non-blocked versions.

For comparing the different algorithms we ran an extensive suite of experiments across a range of ranks (of matrices W and H) and degree of noise in the dataset. The basic methodology was: 1) for each given value of the rank K of the NMA, we generated random matrices W_0 and H_0 and formed the dataset A; 2) we added uniform random or Poisson noise N to A so that (after scaling) $\|(\mathsf{A}+\mathsf{N}) - \mathsf{A}\|_{\mathrm{F}} \,/\, \|\mathsf{A}\|_{\mathrm{F}}$ corresponds to a given level of distortion (2%, 5%, 10%, or 20%); 3) ran all algorithms until the relative change in iterates fell below 10^{-3} or a running time threshold was exceeded.

In all the experiments reported we started each algorithm with the *same* random initialization. The number of blocks used by our algorithms was roughly chosen based on the average number of rows falling in each subset. For e.g., when $m = 1000$, then usually between 2–10 subsets sufficed, and for $m = 3000$ selecting between 20–50 subsets led to better results. The iterative steps (3.2), (3.4), or (4.1) were run between 1 to 4 times.

5.1. Least-squares NMA

Our first results are on the basic LSNMA algorithm [12] that we compare to BI-LSNMA. Figure 2 compares the objective function values on a 1000×1000 matrix for varying ranks of decomposition across a range of distortion levels. We can see that BI-LSNMA performs better than the basic Lee & Seung algorithm, and these differences become more significant as the rank of the approximation is increased. Note that we ran both algorithms for the *same* amount of time, whereby even when our algorithm is only slightly better than the Lee& Seung algorithm, we do not lose anything.

5.2. KL-Divergence NMA

In Figure 3 proceeding row-wise, the figures show objective function values for different noise levels in the data as the rank of approximation is varied for a 1000×1000 input matrix. BI-KLNMA consistently outperforms KLNMA, and this improvement becomes substantial as the rank of approximation is increased. For example, in Figure 3 for rank 320 decompositions, BI-KLNMA yields objective function values up to almost 3 times smaller than the Lee/Seung KLNMA algorithm, in the same amount of running time.

5.3. Bregman-divergence NMA

We now illustrate our algorithm for Bregman-Divergence NMA by showing results for $D_\phi(\mathsf{A};\mathsf{WH})$

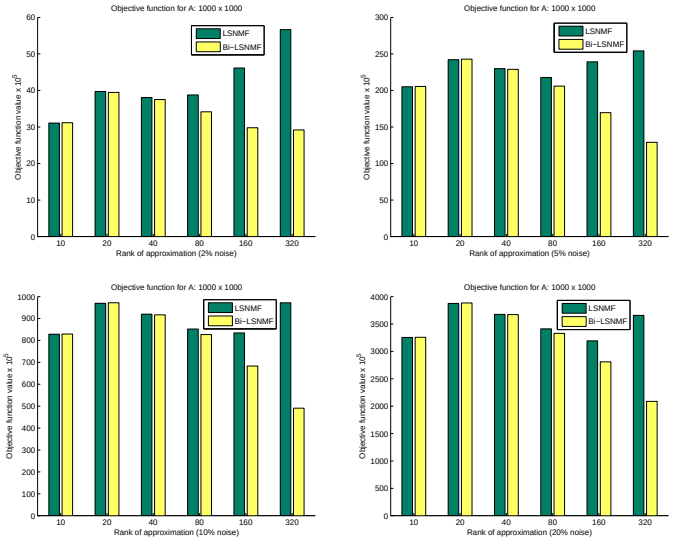

Figure 2. LSNMA **vs.** BI-LSNMA **for a matrix of size** 1000×1000 **with varying noise-levels and rank of approximation. Both algorithms were run for** $\approx$ **60 seconds each.**

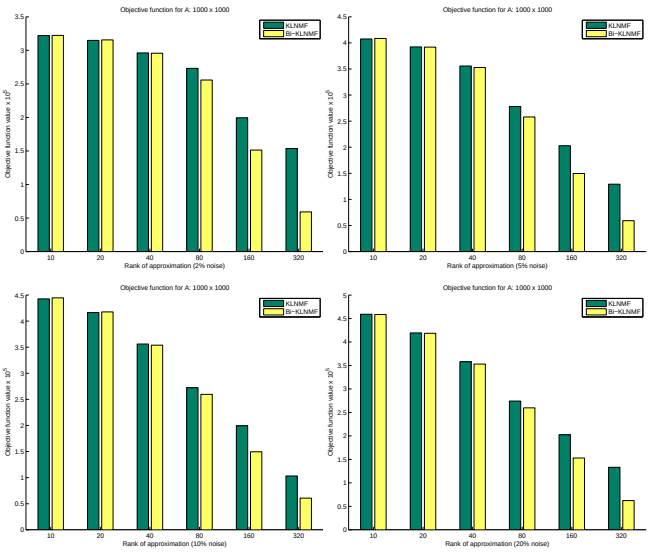

Figure 3. KLNMA **vs.** BI-KLNMA **for a matrix of size** 1000×1000 **with varying noise-levels and rank of approximation. Both algorithms were run for** $\approx$ **200 seconds.**

corresponding to $\phi = -\log x$. With this choice of ϕ, the updates (4.1) simplify to become

$$h_j^{(k,s+1)} = h_j^{(k,s)} \frac{\sum_{i \in S_s} w_{ij} [\mathsf{W}\mathsf{h}^{(k,s)}]_i^{-2} a_i}{\sum_{i \in S_s} w_{ij} [\mathsf{W}\mathsf{h}^{(k,s)}]_i^{-1}}.$$

We also specialized the generic Bregman-NMA code of [6] to take advantage of similar simplifications.

Figure 4 shows experiments on input data matrices of size 1000×1000 with varying ranks and noise levels for the associated NMA. In the figure, for lower rank approximations, both the blocked and unblocked algorithms perform almost the same. The differences for higher rank approximations are overshadowed due to the bad scaling of the y-axis. To highlight the differences better, we summarize the objective function values for the rank ≥ 80 approximations in Table 1, which show that BI-BREGNMA actually yields objective function values twice as good as BREGNMA.

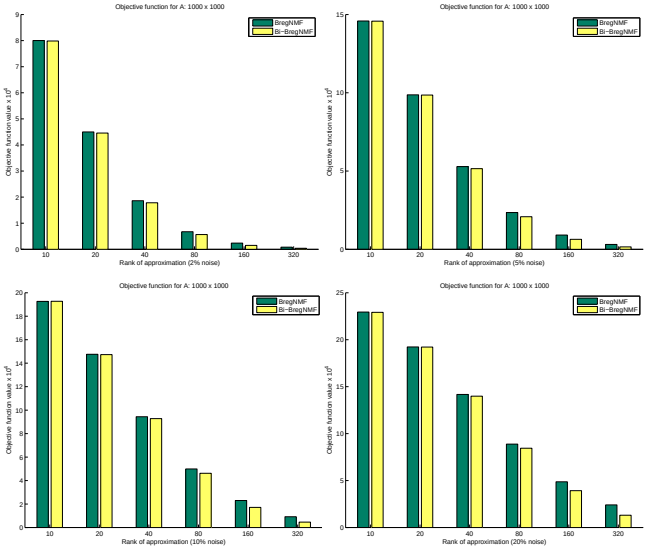

Figure 4. BREGNMA **vs.** BI-BREGNMA **for a matrix of size** 1000×1000 **with varying noise-levels and ranks. Both algorithms were run for** $\approx$ **60 seconds.**

2%	5%	10%	20%
6.75 / **5.65**	23.49 / **20.80**	49.89 / **46.20**	88.84 / **84.42**
2.37 / **1.50**	9.12 / **6.36**	23.03 / **17.17**	48.73 / **39.27**
0.82 / **0.40**	3.11 / **1.53**	9.22 / **4.62**	24.11 / **13.09**

Table 1. Objective function values ($\times 10^3$) for ranks 80, 160, and 320 (row-wise) and noise-levels (column-wise) corresponding to Figure 4. Bold indicates results of BI-BREGNMA**.**

6. Conclusions and Future Work

In this paper we introduced block-iterative algorithms for solving the NMA problem. We showed the generic technique, and applied it to the Frobenius norm, KL-Divergence, and Bregman-divergence NMA problems (of which the first two are special cases). Our use of block-iterative methods was motivated by the successful exploitation of EM techniques for NMA by Lee & Seung [12]. We performed extensive experiments to show the benefits obtained by using a block-iterative scheme, mainly showing that in the same running time, the block-iterative algorithms achieve lower objective function values that the traditional approaches. Furthermore, block-iterative methods are not much more complicated to implement.

The following variants and generalizations of the NMA problem are easy candidates that can be extended via the block-iterative technique.

1. **Multi-factor Problem.** Here we wish to minimize $\mathscr{F}(\mathsf{A}; \mathsf{W}_1\mathsf{W}_2 \ldots \mathsf{W}_T)$, where each $\mathsf{W} \geq 0$. Clearly, by cycling through all the factors and applying a block-iterative method to each subproblem our methods generalize trivially in this case.
2. **Convex and Semi-NMA.** These generalized variations of the NMA problem [7] that drop some non-negativity constraints can also benefit from block-iterative methods. As a first attempt, one can just take the algorithms of [7] and apply the block-iterative techniques to immediately obtain new algorithms.
3. **Non-negative tensor factorization.** NTF is generalization of the NMA idea to tensors, see for e.g., [18, 22], and the block-iterative idea can be easily extended to the EM style algorithms for NTF.

References

[1] H. H. Bauschke and J. M. Borwein. Legendre functions and the method of random Bregman projections. *Journal of Convex Analysis*, 4:27–67, 1997.

[2] M. Berry, M. Browne, A. Langville, P. Pauca, and R. J. Plemmons. Algorithms and Applications for Approximation Nonnegative Matrix Factorization. *Computational Statistics and Data Analysis*, 2006. Preprint.

[3] Y. Censor and S. A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, 1997.

[4] A. Cichocki, R. Zdunek, and S. Amari. Csiszar's Divergences for Non-Negative Matrix Factorization: Family of New Algorithms. In *6th International Conference on Independent Component Analysis and Blind Signal Separation*, volume Springer LNCS 3889, pages 32–39, Charleston SC, USA, 2006.

[5] M. E. Daube-Witherspoon and G. Muehllehner. An Iterative Image Space Reconstruction Algorthm Suitable for Volume ECT. *IEEE Tran. Medical Imaging*, 5(2):61–66, 1986.

[6] I. S. Dhillon and S. Sra. Generalized Nonnegative Matrix Approximations with Bregman Divergences. In *NIPS 18*, Vancouver, Canada, 2006.

[7] C. Ding, T. Li, and M. I. Jordan. Convex and Semi-Nonnegative Matrix Factorization. Technical Report LBNL TR # 60428, LBNL, 2006.

[8] P. J. Green. On Use of the EM for Penalized Likelihood Estimation. *J. Roy. Stat. Soc. Series B*, 52(3):443–452, 1990.

[9] D. Guillamet, J. Vitrià, and B. Schiele. Introducing a weighted nonnegative matrix factorization for image classification. *Pattern Recognition Letters*, 24(14):2447–2454, 2003.

[10] H. M. Hudson and R. S. Larkin. Accelerated image reconstruction using ordered subsets of projection data. *IEEE Tran. Medical Imaging*, 13(4):601–609, 1994.

[11] D. Kim, S. Sra, and I. S. Dhillon. Fast Newton-type methods for the least squares nonnegative matrix approximation problem. In *SIAM Data Mining*, 2007.

[12] D. D. Lee and H. S. Seung. Algorithms for Nonnegative Matrix Factorization. In *Neural Information Processing Systems*, pages 556–562, 2000.

[13] T. Li and C. Ding. The Relationships Among Various Nonnegative Matrix Factorization Methods for Clustering. In *Proc. IEEE Int. Conf. Data Mining*, pages 362–371, 2006.

[14] T. L. Markham. Factorizations of completely positive matrices. *Proceedings of the Cambridge Philosophical Society*, 69:53–58, 1971.

[15] P. Paatero and U. Tapper. Positive Matrix Factorization: A Nonnegative Factor Model with Optimal Utilization of Error Estimates of Data Values. *Environmetrics*, 5(111–126), 1994.

[16] A. R. D. Pierro. A modified Expectation Maximization Algorithm for Penalized Likelihood Estimation in Emission Tomography. *IEEE Tran. Med. Imag.*, 14(1):132–137, 1995.

[17] A. J. Reader and H. Zaidi. Advances in PET Image Reconstruction. *PET Clinics*, 2(2):173–190, April 2007.

[18] A. Shashua and T. Hazan. Non-Negative Tensor Factorization with Applications to Statistics and Computer Vision. In *ICML*, 2005.

[19] L. A. Shepp and Y. Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE Tran. Medical Imaging*, 1:113–122, October 1982.

[20] S. Sra and I. S. Dhillon. Nonnegative Matrix Approximation: Algorithms and Applications. Technical Report TR-06-27, Univ. of Texas at Austin, 2006.

[21] S. Vavasis. On the complexity of nonnegative matrix factorization. arXiv, September 2007. arXiv:0708.4149v2.

[22] M. Welling and M. Weber. Positive tensor factorization. *Pattern Recognition Letters*, 22:1255–1261, 2001.