Algorithms, Initializations, and Convergence for the Nonnegative Matrix Factorization

Russell Albright°, James Cox°, David Duling°, Amy N. Langville†, and Carl D. Meyer*

Abstract

It is well-known that good initializations can improve the speed and accuracy of the solutions of many nonnegative matrix factorization (NMF) algorithms [56]. Many NMF algorithms are sensitive with respect to the initialization of **W** or **H** or both. This is especially true of algorithms of the alternating least squares (ALS) type [55], including the two new ALS algorithms that we present in this paper. We compare the results of six initialization procedures (two standard and four new) on our ALS algorithms. Lastly, we discuss the practical issue of choosing an appropriate convergence criterion.

Key words. nonnegative matrix factorization, alternating least squares, initializations, convergence criterion, image processing, text mining, clustering

AMS subject classifications. 65B99, 65F10, 65C40, 60J22, 65F15, 65F50

° SAS Institute, Inc.,
Cary, NC 27513-2414, USA
{russell.albright, james.cox, david.duling}@sas.com

† Department of Mathematics,
College of Charleston,
Charleston, SC 29424, USA
langvillea@cofc.edu
Phone: (843) 953-8021

* Department of Mathematics,
Center for Research in Scientific Computation,
N. Carolina State University, Raleigh, N.C. 27695-8205, USA
meyer@math.ncsu.edu
Phone: (919) 515-2384
Research supported in part by NSF CCR-ITR-0113121 and NSF DMS 9714811.

1 Introduction

Nonnegative data are pervasive. Consider the following four important applications, each of which give rise to nonnegative data matrices.

- In document collections, documents are stored as vectors. Each element of a document vector is a count (possibly weighted) of the number of times a corresponding term appears in that document. Stacking document vectors one after the other creates a nonnegative term-by-document matrix that represents the entire document collection numerically.
- Similarly, in image collections, each image is represented by a vector, and each element of the vector corresponds to a pixel. The intensity and color of the pixel is given by a nonnegative number, thereby creating a nonnegative pixel-by-image matrix.
- For item sets or recommendation systems, the information for a purchase history of customers or ratings on a subset of items is stored in a non-negative sparse matrix.
- In gene expression analysis, gene-by-experiment matrices are formed from observing the gene sequences produced under various experimental conditions.

These are but four of the many interesting applications that create nonnegative data matrices (and tensors) [5].

Three common goals in mining information from such matrices are: (1) to automatically cluster similar items into groups, (2) to retrieve items most similar to a user's query, and (3) identify interpretable critical dimensions within the collection. For the past decade, a technique called Latent Semantic Indexing (LSI) [4], originally conceived for the information retrieval problem and later extended to more general text mining problems, was a popular means of achieving these goals. LSI uses a well-known factorization of the term-by-document matrix, thereby creating a low rank approximation of the original matrix. This factorization, the singular value decomposition (SVD) [22, 40], is a classic technique in numerical linear algebra.

The SVD is easy to compute and works well for points (1) and (2) above, but not (3). The SVD does not provide users with any interpretation of its mathematical factors or why it works so well. A common complaint from users is: *do the SVD factors reveal anything about the data collection?* Unfortunately, for the SVD, the answer to this question is no, as explained in the next section. However, an alternative and much newer matrix factorization, known as *the nonnegative matrix factorization (NMF), allows the question to be answered affirmatively.* As a result, it can be shown that the NMF works nearly as well as the SVD on points (1) and (2), and further, can also achieve goal (3).

Most examples and applications of the NMF in this paper refer to text mining because this is the area with which we are most familiar. However, the phrase "term-by-document matrix" which we will use frequently throughout this paper can just as easily be replaced with gene-by-observation matrix, purchase-by-user matrix, etc., depending on the application area.

2 Low Rank Approximations

Applications, such as text processing, data mining, and image processing, store pertinent information in a huge matrix. This matrix $\mathbf{A}$ is large, sparse, and often times nonnegative. In the last few decades, researchers realized that the data matrix could be replaced with a related matrix, of much lower rank. The low rank approximation to the data matrix $\mathbf{A}$ brought several advantages. The rank-k approximation, denoted $\mathbf{A}_k$, sometimes required less storage than $\mathbf{A}$. But most importantly, the low rank matrix seemed to give a much cleaner, more efficient representation of the relationship between data elements. The low rank approximation identified the most essential components of the data by ignoring inessential components attributed to noise, pollution, or inconsistencies. Several low rank approximations are available for a given matrix: QR, URV, SVD, SDD, PCA, ICA, NMF, CUR, etc. [30, 40, 55, 18]. In this section, we focus on two such approximations, the SVD and the NMF, that have been applied to data mining problems.

2.1 The Singular Value Decomposition

In 1991, Susan Dumais [20] used the singular value decomposition (SVD) to build a low rank approximation to the term-by-document matrix of information retrieval. In fact, to build a rank-k approximation $\mathbf{A}_k$ to the rank r term-by-document matrix $\mathbf{A}$, simply use the k most significant singular components, where $k < r$. That is,

$$\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^T,$$

where σ_i is the i^{th} singular value of $\mathbf{A}$, and $\mathbf{u}_i$ and $\mathbf{v}_i^T$ are the corresponding singular vectors [22]. The technique of replacing $\mathbf{A}$ with the truncated $\mathbf{A}_k$ is called Latent Semantic Indexing (LSI) because the low rank approximation reveals meanings and connections between documents that were hidden, or latent, in the original noisy data matrix $\mathbf{A}$.

Mathematically, the truncated SVD has one particularly appealing property: of all possible rank-k approximations, $\mathbf{A}_k$ is the best approximation in the sense that $\|\mathbf{A} - \mathbf{A}_k\|_F$ is as small as possible [4, 6]. Thus, the truncated SVD provides a nice baseline against which all other low-rank approximations can be judged for quantitative accuracy. This optimality property is also nice in practice. Algorithms for computing the k most significant singular components are fast, accurate, well-defined, and robust [2, 4, 22]. Two different algorithms will produce the same results up to roundoff error. Such uniqueness and computational robustness are comforting. Another advantage of the truncated SVD concerns building successive low rank approximations. Once $\mathbf{A}_{100}$ has been computed, no further computation is required if, for example, for sensitivity analysis or comparison purposes, other *lower* rank approximations are needed. That is, once $\mathbf{A}_{100}$ is available, then $\mathbf{A}_k$ is available for any $k \leq 100$.

LSI and the truncated SVD dominated text mining research in the 1990s [1, 3, 4, 7, 6, 8, 10, 11, 15, 20, 26, 28, 27, 36, 59, 61, 62]. However, LSI is not perfect. For instance, while it first appeared that the low rank approximation $\mathbf{A}_k$ would save storage over the original matrix $\mathbf{A}$, experiments showed that this was not the case. $\mathbf{A}$ is generally very sparse for text mining problems because only a small subset of the terms in the collection are used in any particular document. No matter how sparse the original term-by-document matrix is, the truncated SVD produces singular components that are almost always completely dense. In many cases, $\mathbf{A}_k$ can require more (sometimes much more) storage than $\mathbf{A}$.

Furthermore, $\mathbf{A}$ is always a nonnegative matrix, yet the singular components are mixed in sign. The SVD's loss of the nonnegative structure of the term-by-document matrix means that the factors of the truncated SVD provide no interpretability. To understand this statement, consider a particular document vector, say, column 1 of $\mathbf{A}$. The truncated SVD represents document 1, $\mathbf{A}_1$, as

$$\mathbf{A}_1 \approx \sigma_1 v_{11} \begin{pmatrix} \vdots \\ \mathbf{u}_1 \\ \vdots \end{pmatrix} + \sigma_2 v_{12} \begin{pmatrix} \vdots \\ \mathbf{u}_2 \\ \vdots \end{pmatrix} + \cdots + \sigma_k v_{1k} \begin{pmatrix} \vdots \\ \mathbf{u}_k \\ \vdots \end{pmatrix},$$

which reveals that document 1 is a linear combination of the singular vectors $\mathbf{u}_i$, also called the basis vectors. The scalar weight $\sigma_i v_{1i}$ represents the contribution of basis vector i in document 1. Unfortunately, the mixed signs in $\mathbf{u}_i$ and $\mathbf{v}_i$ preclude interpretation.

Clearly, the interpretability issues with the SVD's basis vectors are caused by the mixed signs in the singular vectors. Thus, researchers proposed an alternative low rank approximation that maintained the nonnegative structure of the original term-by-document matrix. As a result, the nonnegative matrix factorization (NMF) was created [35, 45]. The NMF replaces the role played by the singular value decomposition

(SVD). Rather than factoring $\mathbf{A}$ as $\mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T$, the NMF factors $\mathbf{A}$ as $\mathbf{W}_k\mathbf{H}_k$, where $\mathbf{W}_k$ and $\mathbf{H}_k$ are nonnegative.

2.2 The Nonnegative Matrix Factorization

Recently, the nonnegative matrix factorization (NMF) has been used to create a low rank approximation to $\mathbf{A}$ that contains nonnegative factors called $\mathbf{W}$ and $\mathbf{H}$. The NMF of a data matrix $\mathbf{A}$ is created by solving the following nonlinear optimization problem.

$$\begin{aligned} \min \|\mathbf{A}_{m\times n} \quad &- \quad \mathbf{W}_{m\times k}\mathbf{H}_{k\times n}\|_F^2, \\ s.t. \quad & \quad \mathbf{W} \geq \mathbf{0}, \\ & \quad \mathbf{H} \geq \mathbf{0}. \end{aligned} \tag{1}$$

The Frobenius norm is often used to measure the error between the original matrix $\mathbf{A}$ and its low rank approximation $\mathbf{WH}$, but there are other possibilities [17, 35, 43]. The rank of the approximation, k, is a parameter that must be set by the user.

The NMF is used in place of other low rank factorizations, such as the singular value decomposition (SVD) [40], because of its two primary advantages: storage and interpretability. Due to the nonnegativity constraints, the NMF produces a so-called "additive parts-based" representation [35] of the data. One consequence of this is that the factors $\mathbf{W}$ and $\mathbf{H}$ are generally naturally sparse, thereby saving a great deal of storage when compared with the SVD's dense factors.

The NMF also has impressive benefits in terms of interpretation of its factors, which is, again, a consequence of the nonnegativity constraints. For example, consider a text processing application that requires the factorization of a term-by-document matrix $\mathbf{A}_{m\times n}$. In this case, k can be considered the number of (hidden) topics present in the document collection. In this case, $\mathbf{W}_{m\times k}$ becomes a term-by-topic matrix whose columns are the NMF basis vectors. The nonzero elements of column 1 of $\mathbf{W}$ (denoted $\mathbf{W}_1$), which is sparse and nonnegative, correspond to particular terms. By considering the highest weighted terms in this vector, one can assign a label or topic to basis vector 1. Figure 1 shows four basis vectors for one particular term-by-document matrix, the `medlars` dataset of medical abstracts, available at `http://www.cs.utk.edu/~lsi/`. For those familiar with the domain of this dataset, the NMF allows users the ability to interpret the basis vectors. For instance, a user might attach the label "heart" to basis vector $\mathbf{W}_1$ of Figure 1. Similar interpretation holds for the other factor $\mathbf{H}$. $\mathbf{H}_{k\times n}$ becomes a topic-by-document matrix with sparse nonnegative columns. Element j of column 1 of $\mathbf{H}$ measures the strength to which topic j appears in document 1.

Another fascinating application of the NMF is image processing. Figure 2 clearly demonstrates two advantages of the NMF over the SVD. First, notice that the NMF basis vectors, represented as individual blocks in the $\mathbf{W}$ matrix, are very sparse (i.e., there is much white space). Similarly, the weights, represented as individual blocks in the $\mathbf{H}_i$ vector, are also sparse. On the other hand, the SVD factors are nearly completely dense. Second, the basis vectors of the NMF, in the $\mathbf{W}$ matrix, have a nice interpretation, as individual components of the structure of the face—ears, noses, mouths, hairlines. The SVD basis vectors do not create an additive parts-based representation. In addition, the gains in storage and interpretability do not come at a loss in performance. The NMF and the SVD perform equally well in reconstructing an approximation to the original image.

Of course, the NMF has its disadvantages too. Other popular factorizations, especially the SVD, have strengths concerning uniqueness and robust computation. Yet these become problems for the NMF. There is no unique global minimum for the NMF. The optimization problem of Equation (2) is convex in either $\mathbf{W}$ or $\mathbf{H}$, but not in both $\mathbf{W}$ and $\mathbf{H}$, which means that the algorithms can only, if at all, guarantee convergence to a local minimum. In practice, NMF users often compare the local minima from several different starting points, using the results of the best local minimum found. However, this is prohibitive on large, realistically-sized problems. Not only will different NMF algorithms (and there are many now

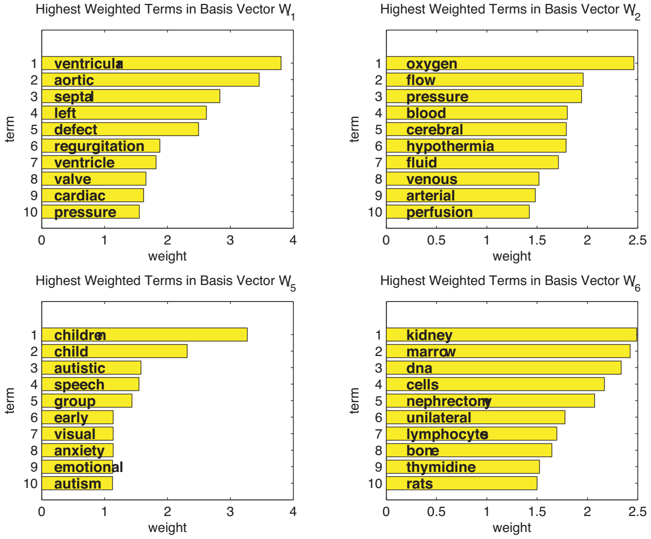

Figure 1: Interpretation of NMF basis vectors on `medlars` dataset

[5]) produce different NMF factors, the same NMF algorithm, run with slightly different parameters, can produce different NMF factors.

2.3 Summary of SVD vs. NMF

We pause to briefly summarize the advantages of these two competing low rank approximations. The properties and advantages of the SVD include: (1) an optimality property; the truncated SVD produces the best rank-k approximation (in terms of squared distances), (2) speedy and robust computation, (3) unique factorization; initialization does not affect SVD algorithms, and (4) orthogonality; resulting basis vectors are orthogonal and allow conceptualization of original data as vectors in space. On the other hand, the advantages of NMF are: (1) sparsity and nonnegativity; the factorization maintains these properties of the original matrix, (2) reduction in storage; the factors are sparse, which also results in easier application to new data, and (3) interpretability; the basis vectors naturally correspond to conceptual properties of the data.

The strengths of one approximation become the weaknesses of another. The most severe weakness of the NMF are its convergence issues. Unlike the SVD and its unique factorization, there is no unique NMF factorization. Because different NMF algorithms can converge to different local minima (and even this convergence to local minima is not guaranteed), initialization of the algorithm becomes critical. In practice, knowledge of the application area can help guide initialization choices. We will return to such initialization issues in Section 4.

3 ALS Algorithms for the NMF

Several currently popular NMF algorithms [14, 35, 34, 45, 43, 60] do not create sparse factors, which are desired for storage, accuracy, and interpretability reasons. Even with adjustments to create sparse factors, the improved algorithms [24, 34, 46, 53] exhibit an undesirable *locking* phenomenon, as explained below. Thus, in this section, we propose two new NMF algorithms [31], called ACLS and AHCLS, that produce sparse factors and avoid the so-called locking problem.

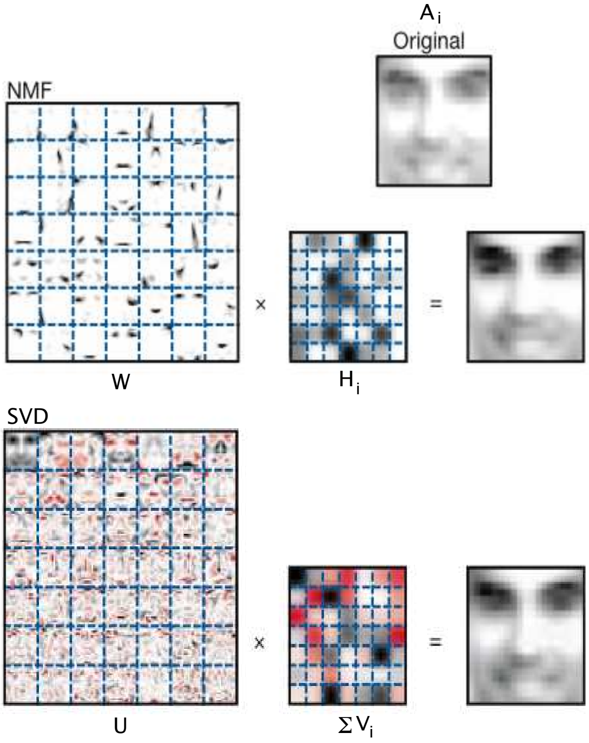

Figure 2: Interpretation of NMF and SVD basis vectors on face dataset, from [35]

Both algorithms are modifications to the simple Alternating Least Squares (ALS) algorithm [45], wherein $\mathbf{W}$ is fixed and $\mathbf{H}$ is computed using least squares, then $\mathbf{H}$ is fixed and $\mathbf{W}$ is computed using least squares, and so on, in alternating fashion. The method of alternating variables is a well-known technique in optimization [42]. One problem with the first ALS algorithm applied to the NMF problem (done by Paatero and Tapper in 1994 [45]) was the lack of sparsity restrictions. To address this, the ACLS algorithm adds a reward for sparse factors of the NMF. The user sets the two parameters λ_H and λ_W to positive values. Increasing these values increases the sparsity of the two NMF factors. However, because there are no upperbounds on these parameters, a user must resort to trial and error to find the best values for λ_H and λ_W. The more advanced AHCLS [31], presented in Section 3.2, provides better sparsity parameters with more intuitive bounds.

3.1 The ACLS Algorithm

The ACLS (Alternating Constrained Least Squares) algorithm is implemented differently than the original ALS algorithm [45] because issues arise at each alternating step, where a constrained least squares problem of the following form

$$\min_{\mathbf{h}_j} \|\mathbf{a}_j - \mathbf{W}\mathbf{h}_j\|_2^2 + \lambda_H \|\mathbf{h}_j\|_2^2 \quad s.t. \quad \lambda_H \geq 0, \mathbf{h}_j \geq \mathbf{0} \tag{2}$$

must be solved. The vectors $\mathbf{a}_j$ and $\mathbf{h}_j$ are columns of $\mathbf{A}$ and $\mathbf{H}$, respectively. Notice that the decision variable $\mathbf{h}_j$ must be nonnegative. There are algorithms specifically designed for this nonnegative constrained least squares problem. In fact, the NNLS algorithm of Lawson and Hanson [9, 33] is so common that it appears as a built-in function in MATLAB. Unfortunately, the NNLS algorithm is very slow, as it is an "active set" method, meaning it can swap only one variable from the basis at a time. Even the faster version of the NNLS algorithm by Bro and de Jong [12] is still not fast enough, and the NNLS step remains the computational bottleneck. As a result, in practice, compromises are made. For example, a standard (unconstrained) least squares step is run [9] and all negative elements in the solution vector are set to 0. This ad-hoc enforcement of nonnegativity, while not theoretically appealing, works quite well in practice. The practical ACLS algorithm is shown below.

```
Practical ACLS Algorithm for NMF
input $\lambda_W$, $\lambda_H$
$\mathbf{W}$ = rand(m,k);   % initialize $\mathbf{W}$ as random dense matrix or use another initialization from Section 4
for i = 1 : maxiter
    (cls)      Solve for $\mathbf{H}$ in matrix equation $(\mathbf{W}^T\mathbf{W} + \lambda_H \mathbf{I})\ \mathbf{H} = \mathbf{W}^T\mathbf{A}$.   % for $\mathbf{W}$ fixed, find $\mathbf{H}$
    (nonneg)   Set all negative elements in $\mathbf{H}$ to 0.
    (cls)      Solve for $\mathbf{W}$ in matrix equation $(\mathbf{H}\mathbf{H}^T + \lambda_W \mathbf{I})\ \mathbf{W}^T = \mathbf{H}\mathbf{A}^T$.   % for $\mathbf{H}$ fixed, find $\mathbf{W}$
    (nonneg)   Set all negative elements in $\mathbf{W}$ to 0.
end
```

3.2 The AHCLS Algorithm

ACLS uses a crude measure $\|\mathbf{x}\|_2^2$ to approximate the sparsity of a vector $\mathbf{x}$. The AHCLS replaces this with a more sophisticated measure, $spar(\mathbf{x})$, which was invented by Hoyer [25].

$$spar(\mathbf{x}_{n\times 1}) = \frac{\sqrt{n} - \|\mathbf{x}\|_1/\|\mathbf{x}\|_2}{\sqrt{n} - 1}$$

In AHCLS (Alternating Hoyer-Constrained Least Squares), the user defines two scalars α_W and α_H in addition to λ_H and λ_W of ACLS. For AHCLS, the two additional scalars $0 \leq \alpha_W, \alpha_H \leq 1$ represent a user's desired sparsity in each column of the factors. These scalars, because they range from 0 to 1, match nicely with a user's notion of sparsity as a percentage. Recall that $0 \leq \lambda_W, \lambda_H \leq \infty$ are positive weights associated with the penalties assigned to the density of $\mathbf{W}$ and $\mathbf{H}$. Thus, in AHCLS, they measure how important it is to the user that $spar(\mathbf{W}_{j*}) = \alpha_W$ and $spar(\mathbf{H}_{j*}) = \alpha_H$. Our experiments show that AHCLS does a better job of enforcing sparsity than ACLS does. And the four AHCLS parameters are easier to set. For example, as a guideline, we recommend $0 \leq \lambda_W, \lambda_H \leq 1$, with of course, $0 \leq \alpha_W, \alpha_H \leq 1$. The practical AHCLS algorithm, using matrix systems and ad-hoc enforcement of negativity, is below. $\mathbf{E}$ is the matrix of all ones.

```
Practical AHCLS Algorithm for NMF
input $\lambda_W$, $\lambda_H$, $\alpha_W$, $\alpha_H$ $\mathbf{W}$ = rand(m,k);   % initialize $\mathbf{W}$ as random dense matrix or use another initialization
from Section 4
$\beta_H = ((1-\alpha_H)\sqrt{k} + \alpha_H)^2$
$\beta_W = ((1-\alpha_W)\sqrt{k} + \alpha_W)^2$
for i = 1 : maxiter
    (hcls)     Solve for $\mathbf{H}$ in matrix equation $(\mathbf{W}^T\mathbf{W} + \lambda_H \beta_H \mathbf{I} - \lambda_H \mathbf{E})\ \mathbf{H} = \mathbf{W}^T\mathbf{A}$.
    (nonneg)   Set all negative elements in $\mathbf{H}$ to 0.
    (hcls)     Solve for $\mathbf{W}$ in matrix equation $(\mathbf{H}\mathbf{H}^T + \lambda_W \beta_W \mathbf{I} - \lambda_W \mathbf{E})\ \mathbf{W}^T = \mathbf{H}\mathbf{A}^T$.
    (nonneg)   Set all negative elements in $\mathbf{W}$ to 0.
end
```

3.3 Advantages and Disadvantages of ACLS and AHCLS

3.3.1 Speed

These algorithms have many advantages. For instance, rather than computing the vectors in $\mathbf{H}$ column by column (as is done in [53]), thereby solving sequential least squares problems of the form of Equation (2), one matrix system solve can be executed. Further, because each CLS step solves a *small* $k \times k$ matrix system, ACLS and AHCLS are the fastest NMF algorithms available (and faster than current truncated SVD algorithms). See Section 3.4 for comparative run times. They converge quickly and give very accurate NMF factors.

3.3.2 Sparsity

Only $\mathbf{W}$ must be initialized, and sparsity is incorporated for both NMF factors. We believe that avoidance of the so-called *locking* phenomenon is one reason why the class of ALS algorithms works well in practice. Nearly all other NMF algorithms, especially those of the multiplicative update class [24, 25, 35, 34, 47, 52, 53], lock elements when they become 0. That is, during the iterative process, once an element in either $\mathbf{W}$ or $\mathbf{H}$ becomes 0, it must remain 0. For the basis vectors in the text mining problem, which are stored in $\mathbf{W}$, this means that in order to improve the objective function, the algorithm can only remove terms from, not add terms to, topic basis vectors. As a result, once the algorithm starts down a path toward a particular topic vector, it must continue in that direction. On the other hand, ALS algorithms do not *lock* elements, and thus provide greater flexibility, allowing them to escape from a path heading towards a poor local minimum.

3.3.3 Convergence

It has been proven that ALS algorithms will converge to a fixed point, but this fixed point may be a local extrema or a saddle point [21, 23, 38]. The ACLS and AHCLS algorithms with properly enforced nonnegativity, for example, by the NNLS algorithm, are known to converge to a local minimum [17, 38]. However, our ad-hoc enforcement of nonnegativity, which drastically speeds up the algorithm (and improves sparsity), means there are no proofs claiming convergence to a local minimum; saddle points are now possible. (Actually, this is not so damning for our two ALS algorithms because most NMF algorithms suffer this same problem. The few NMF algorithms believed to guarantee convergence to a local minimum have been proven otherwise [21, 23, 38].) Our experiments [31, 32] and others [29, 44, 45, 43, 48] have shown that the ALS fixed points can be superior to the results of other NMF algorithms.

3.3.4 Nonnegativity

Clearly, ad-hoc enforcement of nonnegativity is theoretically unattractive. There are some alternatives to this ad-hoc enforcement of nonnegativity. For instance, one could convert from an alternating least squares approach to an alternating linear programming approach, whereby nonnegativity of variables is enforced in a natural way by the simple constraints of the linear programming formulation. Yet, this has the same problem as the NNLS algorithm, lengthy execution time. A second alternative to ad-hoc enforcement of nonnegativity is to add negativity penalties in the form of logarthmic functions to the NMF objective function [39]. This is a focus of future work.

3.4 Numerical Experiments

Figure 3 compares our ACLS and AHCLS algorithms with the popular Lee-Seung mean squared error algorithm [35] and the GDCLS algorithm [53]. We use our own implementation of GDCLS, which is much faster than the implementation presented in [53]. The speed improvement results from our use of one matrix system rather than serial vector systems to solve the CLS step. This implementation trick was described above for the ACLS and AHCLS algorithms.

To create Figure 3, we used the `medlars` dataset of medical abstracts and the `cisi` dataset of library science abstracts. These figures clearly show how the ALS-type algorithms outperform the Lee-Seung multiplicative update algorithms in terms of accuracy and speed. While the ALS-type algorithms provide similar accuracy to the GDCLS algorithm, they are much faster. This speed advantage continues to hold for much larger collections like the `reuters10` collection, used in Section 4. (Details on the `reuters10` dataset appear in Section 4.3.) On average the ACLS and AHCLS algorithms require roughly 2/3 the time of the GDCLS algorithm. Figure 3 also reports the error in the optimal rank-10 approximation required by the SVD. Notice how close all NMF algorithms come to the optimal factorization error. Also, notice that ACLS and AHCLS require less time than the SVD to produce such good, sparse, nonnegative factorizations.

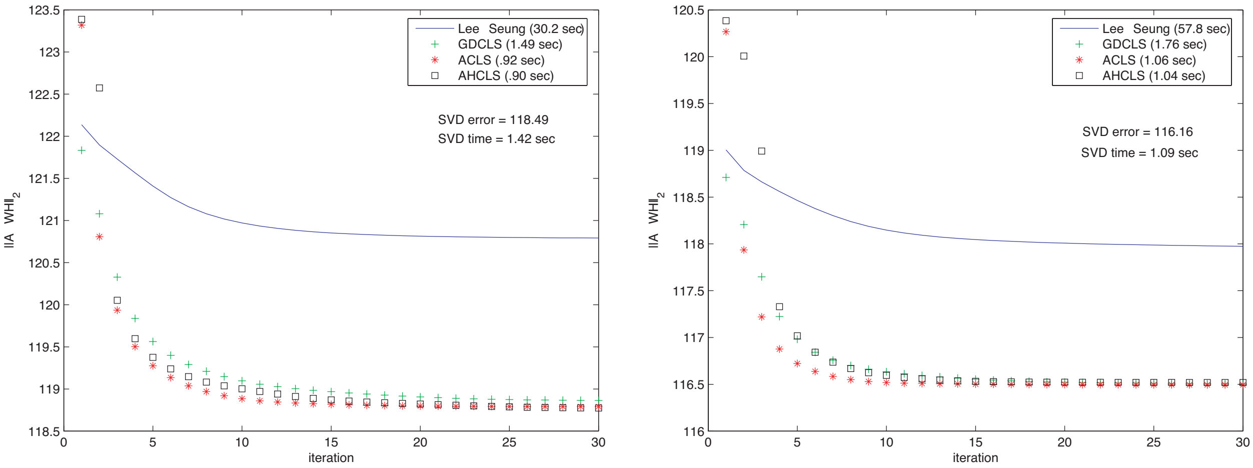

Figure 3: Accuracy and Run-times of NMF Algorithms on `medlars` (left) and `cisi` (right) datasets

4 Initializations

All NMF algorithms are iterative and it is well-known that they are sensitive to the initialization of **W** and **H** [56]. Some algorithms require that both **W** and **H** be initialized [24, 25, 35, 34, 46], while others require initialization of only **W** [45, 43, 52, 53]. *In all cases, a good initialization can improve the speed and accuracy of the algorithms, as it can produce faster convergence to an improved local minimum* [55]. A good initialization can sidestep some of the convergence problems mentioned above, which is precisely why they are so important. In this section, we compare several initialization procedures (two old and four new) by testing them on the ALS algorithms presented in Section 3. We choose to use the ACLS and AHCLS algorithms because they produce sparse accurate factors and require about the same time as the SVD. Most other NMF algorithms require much more time than the SVD, often times orders of magnitude more time.

4.1 Two Existing Initializations

Nearly all NMF algorithms use simple *random initialization*, i.e., **W** and **H** are initialized as dense matrices of random numbers between 0 and 1. It is well-known that random initialization does not generally provide

Table 1: Initialization Methods for the NMF

Name	Proposed by	Pros	Cons
Random	Lee, Seung [34]	easy, cheap to compute	dense matrices, no intuitive basis
Centroid	Wild et al. [56]	reduces # NMF iterations, firm, intuitive foundation	expensive, must run clustering algorithm on cols of **A**
SVD-Centroid	Langville [32]	inexpensive, reduces # NMF iterations	SVD factor **V** must be available
Random Acol	Langville [31]	cheap, sparse matrices built from original data	only slight decrease in number of NMF iterations
Random **C**	Langville adapts from Drineas [18]	cheap, sparse	not very effective
Co-occurrence	Langville adapts from Sandler [50]	uses term-term similarities	large, dense co-occurrence matrix, very expensive computation

a good first estimate for NMF algorithms [55], especially those of the ALS-type of [13, 37, 49, 51]. Wild et al. [56, 57, 58] have shown that the *centroid initialization*, built from the centroid decomposition [16] is a much better alternative to random initialization. Unfortunately, this decomposition is expensive as a preprocessing step for the NMF. Another advantage of ALS algorithms, such as our ACLS and AHCLS, is that they only require initialization of $\mathbf{W}$. In ALS algorithms, once $\mathbf{W}^{(0)}$ is known, $\mathbf{H}^{(0)}$ is computed quickly by a least squares computation. As a result, we only discuss techniques for computing a good $\mathbf{W}^{(0)}$.

4.2 Four New Initializations

Some text mining software produces the SVD factors for other text tasks. Thus, in the event that the SVD factor $\mathbf{V}$ is available, we propose a *SVD-centroid initialization* [32], which initializes $\mathbf{W}$ with a centroid decomposition of the low dimensional SVD factor $\mathbf{V}_{n\times k}$ [54]. While the centroid decomposition of $\mathbf{A}_{m\times n}$ can be too time-consuming, the centroid decomposition of $\mathbf{V}$ is fast because $\mathbf{V}_{n\times k}$ is much smaller than $\mathbf{A}_{m\times n}$. When the SVD factors are not available, we propose a very inexpensive procedure called *random Acol initialization*. Random Acol forms an initialization of each column of the basis matrix $\mathbf{W}$ by averaging p random columns of $\mathbf{A}$. It makes more sense to build basis vectors from the given data, the sparse document vectors themselves, than to form completely dense random basis vectors, as random initialization does. Random Acol initialization is very inexpensive, and lies between random initialization and centroid initialization in terms of performance [31, 32].

We also present two more initialization ideas, one inspired by the $\mathbf{C}$ matrix of the CUR decomposition [18], and another by the term co-occurrence matrix [50]. We refer to these last two methods as *random* $\mathbf{C}$ *initialization* and *co-occurrence initialization*, respectively. The random $\mathbf{C}$ initialization is similar to the random Acol method, except it chooses p columns at random from the longest (in the 2-norm) columns of $\mathbf{A}$, which generally means the densest columns since our text matrices are so sparse. The idea is that these might be more likely to be the centroid centers. The co-occurrence method first forms a term co-occurrence matrix $\mathbf{C} = \mathbf{A}\mathbf{A}^T$. Next, the method for forming the columns of $\mathbf{W}^{(0)}$ described as Algorithm 2 of [50] is applied to $\mathbf{C}$. The co-occurrence method is very expensive for two reasons. First, for text mining datasets, such as `reuters10`, $m >> n$, which means $\mathbf{C} = \mathbf{A}\mathbf{A}^T$ is very large and often very dense too. Second, the algorithm of [50] for finding $\mathbf{W}^{(0)}$ is extremely expensive, making this method impractical. All six initialization methods are summarized in Table 1. The two existing methods appear first, followed by our four suggested methods.

Table 2: Experiments with Initialization Methods for the NMF

Method	Time $\mathbf{W}^{(0)}$	Storage $\mathbf{W}^{(0)}$	Error(0)	Error(10)	Error(20)	Error(30)
Random	.09 sec	726K	4.28%	.28%	.15%	.15%
Centroid	27.72	46K	2.02%	.27%	.18%	.18%
SVD-Centroid	.65†	56K	2.08%	.06%	.06%	.06%
Random Acol*	.05	6K	2.01%	.21%	.16%	.15%
Random $\mathbf{C}^{\circ}$	.11	22K	3.35%	.29%	.20%	.19%
Co-occurrence	3287	45K	3.38%	.37%	.27%	.25%
ACLS time			.37 sec	3.42	6.78	10.29

† provided $\mathbf{V}$ of the SVD is already available

* each column of $\mathbf{W}^{(0)}$ formed by averaging 20 random columns of $\mathbf{A}$

° each column of $\mathbf{W}^{(0)}$ formed by averaging 20 of the longest columns of $\mathbf{A}$

4.3 Initialization Experiments with Reuters10 dataset

The `reuters10` collection is our subset of the Reuters-21578 version of the Reuter's benchmark document collection of business newswire posts. The Reuters-21578 version contains over 20,000 documents categorized into 118 different categories, and is available online.[1] Our subset, the `reuters10` collection, is derived from the set of documents that have been classified into the top ten most frequently occurring categories. The collection contains 9248 documents from the training data of the "ModApte split" (details of the split are also available at the website above).

The numbers reported in Table 2 were generated by applying the alternating constrained least squares (ACLS) algorithm of Section 3 with $\lambda_H = \lambda_W = .5$ to the `reuters10` dataset. The error measure in this table is relative to the optimal rank-10 approximation given by the singular value decomposition. For this dataset, $\|\mathbf{A} - \mathbf{U}_{10}\mathbf{\Sigma}_{10}\mathbf{V}_{10}^T\|_F = 22656$. Thus, for example, the error at iteration 10 is computed as

$$\text{Error}(10) = \frac{\|\mathbf{A} - \mathbf{W}^{(10)}\mathbf{H}^{(10)}\|_F - 22656}{22656}.$$

We distinguish between quantitative accuracy, as reported in Table 2, and qualitative accuracy as reported in Tables 3 through 9. For text mining applications, it is often not essential that the low rank approximation be terribly precise. Often suboptimal solutions are "good enough." After reviewing Tables 3–9, it is easy to see why some initializations give better accuracy and converge more quickly. They start with basis vectors in $\mathbf{W}^{(0)}$ that are much closer to the best basis vectors found, as reported in Table 3, which was generated by using the basis vectors associated with the best global minimum for the `reuters10` dataset, found by using 500 random restarts. In fact, the relative error for this global minimum is .009%, showing remarkable closeness to the optimal rank-10 approximation. By comparing each subsequent table with Table 3, it's clear why one initialization method is better than another. The best method, SVD-centroid initialization, starts with basis vectors very close to the "optimal" basis vectors of Table 3. On the other hand, random and random Acol initialization are truly random. Nevertheless, random Acol does maintain one clear advantage over random initialization as it creates a very sparse $\mathbf{W}^{(0)}$. The Random C and co-occurrence initializations suffer from lack of diversity. Many of the longest documents in the `reuters10` collection appear to be on similar topics, thus, not allowing $\mathbf{W}^{(0)}$ to cover many of the reuters topics.

Because the algorithms did not produce the "wheat" vector always in column one of $\mathbf{W}$, we have reordered the resulting basis vectors in order to make comparisons easier. We also note that the nonnegative

[1] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

Table 3: Basis vectors of $\mathbf{W}^{(50)}$ from *Best Global Minimum* found for reuters10

$\mathbf{W}_1^{(50)}$	$\mathbf{W}_2^{(50)}$	$\mathbf{W}_3^{(50)}$	$\mathbf{W}_4^{(50)}$	$\mathbf{W}_5^{(50)}$	$\mathbf{W}_6^{(50)}$	$\mathbf{W}_7^{(50)}$	$\mathbf{W}_8^{(50)}$	$\mathbf{W}_9^{(50)}$	$\mathbf{W}_{10}^{(50)}$
tonne	billion	share	stg	mln-mln	gulf	dollar	oil	loss	trade
wheat	year	offer	bank	cts	iran	rate	opec	profit	japan
grain	earn	company	money	mln	attack	curr.	barrel	oper	japanese
crop	qrtr	stock	bill	shr	iranian	bank	bpd	exclude	tariff
corn	rise	sharehol.	market	net	ship	yen	crude	net	import
agricul.	pct	common	england	avg	tanker	monetary	price	dlrs	reagan
wheat	earn	acquisition		interest	ship	frgn-exch.	oil		trade

Table 4: Basis vectors of $\mathbf{W}^{(0)}$ created by *Random Initialization* for reuters10

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
announce	wpp	formality	bulletin	matthews	dramatic	squibb	wag	cochran	erik
medtec	reflagging	simply	awfully	nyt	boca raton	kuwaiti	oils	mln	support
pac	kwik	moonie	blair	barrel	clever	dacca	hears	barriers	sale oil
purina	tilbury	tmg	fresno	purina	billion	democrat	bwtr	deluxe	direct
mezzanine	capacitor	bushnell	farm	june	bkne	induce	nestle	mkc	wheat
foreign	grain	country	leutwiler	trend	clever	rate	fed.	econ.	aid

matrix factorization did produce basis vectors that cover 8 of the 10 "correct" reuters classifications, which appear on the last line of Table 3. The two missing reuters classifications are corn and grain, both of which are lumped into the first basis vector labeled wheat. This first basis vector does break into two separate vectors, one pertaining to wheat and grain and another to corn when the number of basis vectors is increased from $k = 10$ to $k = 12$. We note that these categories have been notoriously difficult to classify, as previously reported in [19].

Table 5: Basis vectors of $\mathbf{W}^{(0)}$ created by *Centroid Initialization* for reuters10

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
tonne	bank	share	medar	cts	iran	rate	oil	stg	strike
wheat	rate	company	mdxr	mmln	gulf	dollar	trade	bill	port
grain	dollar	offer	mlx	loss	attack	bank	price	take-up	union
corn	billion	pct	mlxx	net	iranian	currency	barrel	drain	seaman
crop	pct	stock	mich	shr	missile	market	japan	mature	worker
agriculture	trade	dlrs	troy	dlrs	tanker	monetary	opec	money	ship

Table 6: Basis vectors of $\mathbf{W}^{(0)}$ created by *SVD-Centroid Initialization* for `reuters10`

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
tonne	billion	share	bank	cts	iran	dollar	oil	loss	trade
wheat	year	offer	money	shr	gulf	rate	barrel	oper	japan
grain	earn	company	rate	mln	attack	curr.	opec	profit	japanese
corn	qrtr	stock	stg	net	iranian	yen	crude	cts	tariff
crop	rise	pct	market	mln-mln	missile	japan	bpd	mln	import
agricul.	pct	common	pct	rev	ship	economic	price	net	country

Table 7: Basis vectors of $\mathbf{W}^{(0)}$ created by *Random Acol Initialization* for `reuters10`

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
mln	fee	agl	mln	mark	loss	official	dlrs	bank	trade
denman	mortg.	tmoc	dlrs	mannes.	mln	piedmont	oper	bancaire	viermetz
dlrs	billion	bank	share	dividend	cts	dollar	billion	austral	mln
ecuador	winley	pct	seipp	mln	maki	interest	loss	neworld	nwa
venezuela	mln	company	billion	dieter	name	tokyo	texaco	datron	cts
revenue	fed	maki	dome	gpu	kato	japanese	pennzoil	share	builder

Table 8: Basis vectors of $\mathbf{W}^{(0)}$ created by *Random* **C** *Initialization* for `reuters10`

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
analyst	dollar	econ.	bank	market	analyst	analyst	analyst	trade	rate
lawson	rate	policy	rate	bank	market	industry	bank	dollar	trade
market	econ.	pct	market	analyst	trade	price	currency	japan	official
trade	mark	cost	currency	price	pct	market	japan	price	bank
sterling	bank	growth	dollar	mark	last	believe	billion	japanese	market
dollar	rise	trade	trade	good	official	last	cut	pct	econ.

Table 9: Basis vectors of $\mathbf{W}^{(0)}$ created by *Co-occurrence Initialization* for `reuters10`

$\mathbf{W}_1^{(0)}$	$\mathbf{W}_2^{(0)}$	$\mathbf{W}_3^{(0)}$	$\mathbf{W}_4^{(0)}$	$\mathbf{W}_5^{(0)}$	$\mathbf{W}_6^{(0)}$	$\mathbf{W}_7^{(0)}$	$\mathbf{W}_8^{(0)}$	$\mathbf{W}_9^{(0)}$	$\mathbf{W}_{10}^{(0)}$
dept.	average	agricul.	national	farmer	rate-x	aver price	plywood	wash.	trade
wheat	pct	wheat	bank	rate-x	natl	average	aqtn	trade	japan
agricul.	rate	tonne	rate	natl	avge	price	aequitron	japan	billion
tonne	price	grain	pct	avge	farmer	yield	medical	official	market
usda	billion	farm	oil	cwt	cwt	billion	enzon	reagan	japanese
corn	oil	dept.	gov.	wheat	wheat	bill	enzon	pct	import

5 Convergence Criterion

Nearly all NMF algorithms use the simplest possible convergence criterion, i.e., run for a fixed number of iterations, denoted `maxiter`. This criterion is used so often because the natural criterion, stop when $\|\mathbf{A}-\mathbf{W}\mathbf{H}\| \leq \epsilon$, requires more expense than most users are willing to expend, even occasionally. Notice that `maxiter` was the convergence criterion used in the ACLS and AHCLS algorithms of Section 3. However, a fixed number of iterations is not a mathematically appealing way to control the number of iterations executed because the most appropriate value for `maxiter` is problem-dependent.

In this section, we use the ACLS algorithm applied to the `cisi` dataset to compare two convergence criterion: the natural but more expensive Frobenius norm measure, and our proposed angular measure, which we describe in the next paragraph. We note that we used the most efficient implementation of the Frobenius measure [5], which exploits the trace form of the Frobenius norm.

$$\|\mathbf{A}-\mathbf{W}\mathbf{H}\|_F^2 = trace(\mathbf{A}^T\mathbf{A}) - 2\,trace(\mathbf{H}^T\mathbf{W}^T\mathbf{A}) + trace(\mathbf{H}^T\mathbf{W}^T\mathbf{W}\mathbf{H}).$$

In this equation $trace(\mathbf{A}^T\mathbf{A})$ is a constant that does not change throughout the iterations, and thus, is only computed once and stored. At each iteration $2\,trace(\mathbf{H}^T\mathbf{W}^T\mathbf{A})$ and $trace(\mathbf{H}^T\mathbf{W}^T\mathbf{W}\mathbf{H})$ must be computed. However, some calculations required by these traces, such as $\mathbf{W}^T\mathbf{A}$ and $\mathbf{W}^T\mathbf{W}$, are already available from the least squares steps, and hence, need not be recomputed.

Our angular convergence measure is moderately inexpensive in storage and computation, and is intuitively appealing. Simply measure the angle θ_i between successive topic vectors, i.e., $\mathbf{W}_i^{(j+1)}$ and $\mathbf{W}_i^{(j)}$ at iterations j and $j+1$. Once $\theta_i \leq \epsilon$ for $i = 1, \ldots, k$, stop because the topic vectors have converged satisfactorily. Mitchell and Burdick [41] have shown that, in a different context, a similar measure converges simultaneously with the expensive convergence criterion based on the objective function, $\|\mathbf{A}-\mathbf{W}\mathbf{H}\|$ [55]. However, Figure 4 clearly shows one problem with the angular convergence measure—it does not maintain continual descent, because the basis vectors compared from one iteration to the next are not required to maintain any fixed column order. After several iterations, the column ordering of the basis vectors in $\mathbf{W}$ is less likely to change, making the angular measure more useful in later iterations of the algorithm. The angular convergence measure is much less expensive to compute than the Frobenius measure, but does require additional storage of the $\mathbf{W}$ matrix from the previous iteration. We note in practice, that regardless of the chosen convergence criterion, it is wise to only compute the measure every five or so iterations after some burn-in period.

The recent 2005 reference by Lin [38] mentioned the related convergence criterion issue of stationarity. The fact that $\|\mathbf{A}-\mathbf{W}\mathbf{H}\|$ (or some similar objective function) levels off does not guarantee stationarity. Lin advocates a stationarity check once an algorithm has stopped. For instance, the stationarity checks of Chu and Plemmons [14] may be used. Lin [38] proposes a convergence criterion, that simultaneously checks for stationarity, and fits nicely into his projected gradients algorithm. We agree that a stationarity check should be conducted on termination.

6 Conclusion

The two new NMF algorithms presented in this paper, ACLS and AHCLS, are some of the fastest available, even faster than truncated SVD algorithms. However, while the algorithms will converge to a stationary point, they cannot guarantee that this stationary point is a local minimum. If a local minimum must be achieved, then we recommend using the results from a fast ALS-type algorithm as the initialization for one of the slow algorithms [38] that do guarantee convergence to a local minimum. In this paper we also presented several alternatives to the common, but poor, initialization technique of random initialization. Lastly, we proposed an alternative stopping criterion that practical implementations of NMF code should consider. The common stopping criterion of running for a fixed number of iterations should be replaced with a criterion that fits the context of the users and their data. For many applications, iterating until

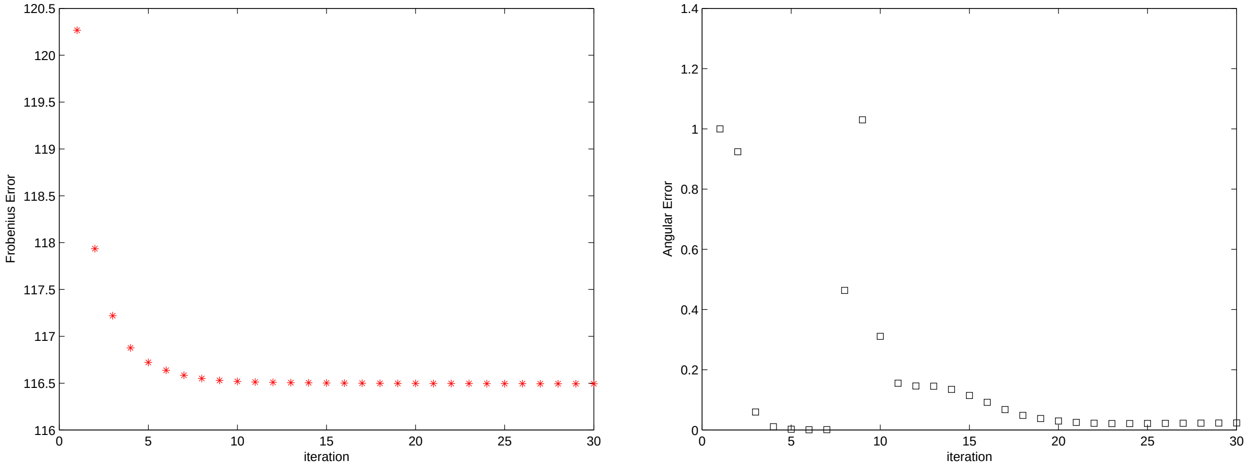

Figure 4: Frobenius convergence measure (left) and angular convergence measure (right) of ACLS Algorithm on `cisi` datasets

$\|\mathbf{A} - \mathbf{WH}\|$ reaches some small level is unnecessary, especially in cases where one is most interested in the qualitative results produced by the vectors in $\mathbf{W}$. In such cases, our proposed angular convergence measure is more appropriate.

References

[1] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, 1999.

[2] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, Philadelphia, 2nd edition, 1994.

[3] Michael W. Berry, editor. *Computational Information Retrieval*. SIAM, Philadelphia, 2001.

[4] Michael W. Berry and Murray Browne. *Understanding Search Engines: Mathematical Modeling and Text Retrieval*. SIAM, Philadelphia, 2nd edition, 2005.

[5] Michael W. Berry, Murray Browne, Amy N. Langville, V. Paul Pauca, and Robert J. Plemmons. Algorithms and applications for the nonnegative matrix factorization. 2006. submitted.

[6] Michael W. Berry, Zlatko Drmac, and Elizabeth R. Jessup. Matrices, vector spaces and information retrieval. *SIAM Review*, 41:335–62, 1999.

[7] Michael W. Berry and R. D. Fierro. Low-rank orthogonal decompositions for information retrieval applications. *Journal of Numerical Linear Algebra with Applications*, 1(1):1–27, 1996.

[8] Michael W. Berry and Gavin W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37:573–595, 1998.

[9] Ake Bjorck. *Numerical methods for least squares problems*. SIAM, Philadelphia, 1996.

[10] Katarina Blom. *Information retrieval using the singular value decomposition and Krylov subspaces.* PhD thesis, University of Chalmers, January 1999.

[11] Katarina Blom and Axel Ruhe. Information retrieval using very short Krylov sequences. In *Computational Information Retrieval*, pages 41–56, 2001.

[12] Rasmus Bro and Sijmen de Jong. A fast non-negativity constrained linear least squares algorithm. *Journal of Chemometrics*, 11:393–401, 1997.

[13] Donald S. Burdick, Xin M. Tu, Linda B. McGown, and David W. Millican. Resolution of multicomponent fluorescent mixtures by analysis of the excitation-emission-frequency array. *Journal of Chemometrics*, 4:15–28, 1990.

[14] Moody Chu, Fasma Diele, Robert J. Plemmons, and Stefania Ragni. Optimality, computation, and interpretations of nonnegative matrix factorizations. *SIAM Journal on Matrix Analysis*, 2004. submitted.

[15] Anirban Dasgupta, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Variable latent semantic indexing. In *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining.* ACM Press, 2005.

[16] Inderjit S. Dhillon. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1/2):143–175, 2001.

[17] Inderjit S. Dhillon and Suvrit Sra. Generalized nonnegative matrix approximations with Bregman divergences. In *Proceeding of the Neural Information Processing Systems (NIPS) Conference*, Vancouver, B.C., 2005.

[18] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 2006. to appear.

[19] Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 148–55. ACM Press, 1998.

[20] Susan T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments and Computers*, 23:229–236, 1991.

[21] Lorenzo Finesso and Peter Spreij. Approximate nonnegative matrix factorization via alternating minimization. In *Sixteenth International Symposium on Mathematical Theory of Networks and Systems*, Leuven, 2004.

[22] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, 1996.

[23] Edward F. Gonzalez and Yin Zhang. Accelerating the Lee-Seung algorithm for nonnegative matrix factorization. Technical Report TR-05-02, Rice University, March 2005.

[24] Patrik O. Hoyer. Non-negative sparse coding. In *Neural Networks for Signal Processing XII (Proc. IEEE Workshop on Neural Networks for Signal Processing)*, pages 557–565, 2002.

[25] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[26] M. K. Hughey and Michael W. Berry. Improved query matching using kd-trees, a latent semantic indexing enhancement. *Information Retrieval*, 2:287–302, 2000.

[27] Eric P. Jiang and Michael W. Berry. Solving total least squares problems in information retrieval. *Linear Algebra and its Applications*, 316:137–156, 2000.

[28] Fan Jiang and Michael L. Littman. Approximate dimension equalization in vector-based information retrieval. In *The Seventeenth International Conference on Machine Learning*, pages 423–430, 2000.

[29] Mika Juvela, Kimmi Lehtinen, and Pentti Paatero. The use of positive matrix factorization in the analysis of molecular line spectra. *Monthly Notices of the Royal Astronomical Society*, 280:616–626, 1996.

[30] Tamara G. Kolda and Dianne P. O'Leary. A semi-discrete matrix decomposition for latent semantic indexing in information retrieval. *ACM Transactions on Information Systems*, 16:322–346, 1998.

[31] Amy N. Langville. Algorithms for the nonnegative matrix factorization in text mining, April 2005. Slides from SAS Meeting.

[32] Amy N. Langville. Experiments with the nonnegative matrix factorization and the reuters10 dataset, February 2005. Slides from SAS Meeting.

[33] Charles L. Lawson and Richard J. Hanson. *Solving Least Squares Problems*. SIAM, 1995.

[34] D. Lee and H. Seung. Algorithms for Non-Negative Matrix Factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

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

[36] Todd A. Letsche and Michael W. Berry. Large-scale information retrieval with LSI. *Informatics and Computer Science*, pages 105–137, 1997.

[37] Shousong Li and Paul J. Gemperline. Eliminating complex eigenvectors and eigenvalues in multiway analyses using the direct trilinear decomposition method. *Journal of Chemometrics*, 7:77–88, 1993.

[38] Chih-Jen Lin. Projected gradient methods for non-negative matrix factorization. Technical Report Information and Support Services Tech. Report ISSTECH-95-013, Department of Computer Science, National Taiwan University, 2005.

[39] Jianhang Lu and Laosheng Wu. Technical details and programming guide for a general two-way positive matrix factorization algorithm. *Journal of Chemometrics*, 18:519–525, 2004.

[40] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, 2000.

[41] Ben C. Mitchell and Donald S. Burdick. Slowly converging PARAFAC sequences: Swamps and two-factor degeneracies. *Journal of Chemometrics*, 8:155–168, 1994.

[42] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.

[43] Pentti Paatero. Least squares formulation of robust non-negative factor analysis. *Chemometrics and Intelligent Laboratory Systems*, 37:23–35, 1997.

[44] Pentti Paatero. The multilinear engine—a table-driven least squares program for solving multilinear problems, including the n-way parallel factor analysis model. *Journal of Computational and Graphical Statistics*, 8(4):1–35, 1999.

[45] Pentti Paatero and U. Tapper. Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:111–126, 1994.

[46] V. Paul Pauca, Jon Piper, and Robert J. Plemmons. Nonnegative matrix factorization for spectral data analysis. 2005.

[47] V. Paul Pauca, Farial Shahnaz, Michael W. Berry, and Robert J. Plemmons. Text Mining Using Non-Negative Matrix Factorizations. In *Proceedings of the Fourth SIAM International Conference on Data Mining, April 22-24*, Lake Buena Vista, FL, 2004. SIAM.

[48] Robert J. Plemmons, 2005. private communication.

[49] E. Sanchez and B. R. Kowalski. Tensorial resolution: A direct trilinear decomposition. *Journal of Chemometrics*, 4:29–45, 1990.

[50] Mark Sandler. On the use of linear programming for unsupervised text classification. In *The Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Chicago, IL, 2005.

[51] R. Sands and Forrest W. Young. Component models for three-way data: an alternating least squares algorithm with optimal scaling features. *Psychometrika*, 45:39–67, 1980.

[52] Farial Shahnaz. A clustering method based on nonnegative matrix factorization for text mining. Master's thesis, University of Tennessee, Knoxville, 2004.

[53] Farial Shahnaz, Michael W. Berry, V.Paul Pauca, and Robert J. Plemmons. Document Clustering Using Nonnegative Matrix Factorization. *Information Processing & Management*, 42(2):373–386, 2006.

[54] David B. Skillicorn, S. M. McConnell, and E.Y. Soong. Handbooks of data ming using matrix decompositions. 2003.

[55] Age Smilde, Rasmus Bro, and Paul Geladi. *Multi-way Analysis*. Wiley, West Sussex, England, 2004.

[56] Stefan Wild. Seeding non-negative matrix factorizations with spherical k-means clustering. Master's thesis, University of Colorado, 2003.

[57] Stefan Wild, James Curry, and Anne Dougherty. Motivating non-negative matrix factorizations. In *Eighth SIAM Conference on Applied Linear Algebra*, Philadelphia, 2003. SIAM.

[58] Stefan Wild, James Curry, and Anne Dougherty. Improving non-negative matrix factorizations through structured initialization. *Journal of Pattern Recognition*, 37(11):2217–2232, 2004.

[59] Dian I. Witter and Michael W. Berry. Downdating the latent semantic indexing model for conceptual information retrieval. *The Computer Journal*, 41(1):589–601, 1998.

[60] Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–273, New York, 2003. ACM Press.

[61] Hongyuan Zha, Osni Marques, and Horst D. Simon. A subspace-based model for information retrieval with applications in latent semantic indexing. *Lecture Notes in Computer Science*, 1457:29–42, 1998.

[62] Xiaoyan Zhang, Michael W. Berry, and Padma Raghavan. Level search schemes for information filtering and retrieval. *Information Processing and Management*, 37:313–34, 2001.