1 Introduction

Low-rank matrix approximation is a widely used tool for data analysis, data compression, and dimensionality reduction, as it describes the underlying structure of input data by computing basis vectors that span a low-dimensional space in which most of the information described by the input data resides. The original matrix may then be well-approximated as a linear combination of these basis vectors.

Techniques for computing a low-rank matrix approximation seek to find a matrix decomposition for a general matrix $A \in \mathbb{R}^{m \times n}$ such that the decomposition has rank $k < \min(m, n)$. Traditionally, this matrix decomposition is computed using the singular value decomposition, so that $A = U \Sigma V^T$, where $U$ and $V$ are orthogonal and $\Sigma$ is a diagonal matrix consisting of the singular values of $A$. Since the singular values are ordered in a decreasing fashion, one can truncate the SVD after the singular values reach some small threshold, so that the resulting decomposition has low rank but is still a good approximation of $A$.

However, SVD-style approaches to computing low-rank matrix approximations have a significant weakness: they allow for both positive and negative components. In many real life data sets, negative terms do not have meaningful interpretations. For instance, astrophysics may involve analyzing non-negative frequencies, and values in a grey-scale image have a non-negative range of 0 to 255.

Non-negative matrix factorization (NMF) is a form of low-rank matrix approximation where both the basis vectors and the weights are constrained to be non-negative. This constraint ensures that input data is only represented as a linear combination of these non-negative basis vectors with non-negative coefficients. More specifically, given $V \in \mathbb{R}^{m \times n}_+$ with large $m$ and $n$, the goal of NMF is to find two matrices $W \in \mathbb{R}^{m \times k}_+$ and $H \in \mathbb{R}^{k \times n}_+$ of low rank $k$ so that $WH$ approximates the matrix $V$, where $W \geq 0$ and $H \geq 0$.

Paatero and Tapper [7] first introduced NMF in 1994, and it was then popularized by Lee and Seung [6] in 1999. NMF has grown in recognition because it extracts sparse and easily interpretable factors. Common applications of NMF include deep learning [4], feature recognition [5], and audio source separation [2, 3, 8, 9].

Feature extraction is crucial for facial recognition. Erichson implements various algorithms to illustrate the impact of NMF vs. SVD on facial feature extraction [1]. When SVD is used, the images represent ‘shadows’ of faces. However, NMF allows for parts-based representations of faces by constraining the data to non-negative pixel values. Figure 1
Figure 1: The figure shows the 16 dominant features extracted. The difference between the NMF and SVD images clearly shows that the features extracted using NMF are easier to interpret. The original figures come from Yale face database B. [1] highlights that features extracted with NMF are more frugal (less data extracted) and more interpretable.

We propose to first explore basic algorithms for computing NMF [6]. Then, we will move on to focusing on improving computational complexity [1]. Finally, we will apply these more efficient algorithms to audio source separation of musical instruments.

2 Background

NMF can be formulated as an optimization problem. Our goal is to minimize $\|V - WH\|^2$ with respect to $W$ and $H$, where $W, H \geq 0$. The function $\|V - WH\|^2$ is convex in only $W$ or only $H$, so we cannot algorithmically find global minima. Instead, we must use numerical optimization to find local minima.

Lee and Seung suggest the following theorem [6].

**Theorem 1.** The Euclidean distance $\|V - WH\|^2$ is non-increasing under the update rules:

$$H_{\alpha\mu} \leftarrow H_{\alpha\mu} \frac{(W^TV)_{\alpha\mu}}{(W^TW)_\alpha} \quad W_{i\alpha} \leftarrow W_{i\alpha} \frac{(VH^T)_{i\alpha}}{(WHH^T)_{i\alpha}}.$$

The Euclidean distance is invariant (under these updates) if and only if $W$ and $H$ are at a stationary point of distance. These update rules are a weighted form of gradient descent.

Lee and Seung split their iterative approach into two stages. First, they fix $W$, which is initialized as a random guess, and update $H$ using the update rule above. Then, they fix $H$ and update $W$ using the respective update rule for $W$, before moving on to the next iteration. To prove convergence of this scheme, Lee and Seung analyze only the first stage. The proof for the second stage follows using the same logic.

To prove convergence of this method [6], the following least squares error must be minimized where each $h_i$ is a column vector of $H$ and each $v_i$ is a column vector of $V$:

$$F(h) = 1/2 \sum_i \left( v_i - \sum_\alpha W_{i\alpha} h_\alpha \right)^2.$$
From the multiplicative update theorem, we get the update rule
\[ h_{i+1} = h_i - K(h_i)^{-1} \nabla F(h_i), \]
where \( K(h_i) \) is the diagonal matrix
\[ K_{ab}(h_i) = \frac{\delta_{ab}(W^TWh_i)_a}{h_{ia}}. \]

Lee and Seung show that the function \( F \) is non-increasing with respect to \( h_i \), with the lower-bound of zero. Therefore, even though the weight (or “step size”) \( K(h_i)^{-1} \) may be large, the iterative scheme will always converge as \( t \) increases.

This approach by Lee and Seung was the first computationally feasible method for computing the NMF. Since then, many new methods have emerged that are able to reduce the computational complexity required to compute \( W \) and \( H \). We will explore some of these more efficient methods in the course of our project.

### 3 Proposed Methodology

We will begin by implementing NMF through Lee’s least-squares update rule [6].

\[
H_{\alpha \mu} \leftarrow H_{\alpha \mu} \frac{(W^TV)_{\alpha \mu}}{(W^TWH)_{\alpha \mu}} \quad W_{i\alpha} \leftarrow W_{i\alpha} \frac{(VH^T)_{i\alpha}}{(WHHT)_{i\alpha}}
\]

Our programming will be done using MATLAB. Lee and Seung comment that this approach is not necessarily computationally efficient, and the works of Gillis [5] and Erichson [1] seek to develop highly efficient solvers, which we will explore.

We will follow the algorithms described in Section 3 of Gillis [5] and Section 3 of Erichson [1] to improve computational efficiency.

Gillis [5] references the Alternating Non-negative Least Squares Algorithm (ANLSA) as a more efficient approach, where the sub-problems in \( W \) and \( H \) are solved exactly. The update rules for \( W \) and \( H \) are given by:

\[
W = \max \left( 0, \frac{VH^T}{HH^T} \right), \quad H^T = \max \left( 0, \frac{V^TW}{WTW} \right),
\]

where \( W \) is computed by ANLSA.

Gillis also introduces the Hierarchical Alternating Least Squares Algorithm, which uses coordinate descent to update each subsequent column of \( W \) separately. For \( l = 1, 2, ..., r \) the update rule is:

\[
W(:,l) \leftarrow \text{argmin}_{W(:,l) \geq 0} \left\| V - \sum_{k \neq l} W(:,l)H(k,:) - W(:,l)H(l,:) \right\|_F
\]
\[
\leftarrow \max \left( 0, \frac{VH(l,:)^T - \sum_{k \neq l} W(:,k)(H(k,:)H(l,:)^T)}{\|H(l,:)\|_2^2} \right).
\]
Lastly, we will implement Erichson’s Randomized Hierarchal Alternating Least Squares Algorithm [1]. First, we rewrite the optimization problem as low-dimensional optimization by replacing the high-dimensional input matrix $V \in \mathbb{R}^{m \times n}$ with the surrogate matrix $B \in \mathbb{R}^{k \times n}$ where we minimize

$$\tilde{f}(\tilde{W}, H) = \| B - \tilde{W}H \|_F^2.$$ 

The non-negativity constraint only applies to the high-dimensional factor matrix $W$ but not necessarily to $\tilde{W}$, where $W = Q\tilde{W}$ where $Q$ is some orthogonal matrix. This optimization problem can be reformulated as minimizing

$$\tilde{J}_j(\tilde{W}(:,j), H(:,j)) = \left\| \left( B - \sum_{i \neq j}^{k} \tilde{W}(:,i)H(:,i) \right) - \tilde{W}(:,j)H(:,j) \right\|_F^2.$$ 

Finally, we will apply these various NMF algorithms to audio source separation. We will use audio files of separate musical instruments shipped with MatLab as test data. We propose to combine these audio files, and then utilize the NMF algorithms to re-separate the instruments. To compare the accuracy of the different algorithms, we will compare the separated audio files to the original, uncombined audio sources using the $\ell^2$ norm. We will also listen to the audio files and subjectively judge which sounds best. We will investigate whether the solutions with the smallest $\ell^2$ error correspond to the best solutions in a qualitative sense. Our goals are to find the most accurate and the most computationally efficient NMF algorithm, and to demonstrate the functionality of these algorithms for audio source separation.

**References**


