4-2020

Nonnegative Matrix Factorization Problem

Junda An

Nonnegative Matrix Factorization Problem

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelors of Science in Mathematics from
The College of William and Mary

by

Junda An

Accepted for Honors
(Honors. no Honors)

Charles R. Johnson

Charles Johnson, Advisor

Gexin Yu

Martin White

Williamsburg. VA

NONNEGATIVE MATRIX FACTORIZATION PROBLEM

Junda An, B.S.

College of William and Mary 2020

The Nonnegative Matrix Factorization (NMF) problem has been widely used to analyze high-dimensional nonnegative data and extract important features. In this paper, I review major concepts regarding NMF, some NMF algorithms and related problems including initialization strategies and near separable NMF. Finally I will implement algorithms on generated and real data to compare their performances.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my advisor Professor Charles Johnson for his continuous support of my undergraduate study and research, and for his motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

Besides my advisor, I would like to thank the rest of my thesis committee: Professor Gexin Yu and Professor Martin White, for their insightful comments and inspiring questions.

Last but not least, I would like to thank my parents: Dewen An and Huichun Zhou, for giving birth to me at the first place and supporting me throughout my life.

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

The Nonnegative Matrix Factorization problem has been extensively studied since [Le Se] proposed simple and useful algorithms to approach this problem in 1999. After they published their work, there have been many algorithms attempting to solve this problem and several variants. Our goal is to state the NMF problem, explain algorithms we know, and introduce some further extensions of NMF.

The NMF is stated as follows. Given an $m \times n$ matrix X and a positive integer $r < min\{m, n\}$, find nonnegative matrices $W \in \mathbb{R}_{+}^{m\times r}$ and $H \in \mathbb{R}_{+}^{r\times n}$ to minimize the function $f(W, H) = \frac{1}{2}||X - WH||_F^2$. X is not equal to WH in most cases, but the product WH is called a nonnegative factorization of X. So, strictly speaking, this problem should be phrased as Nonnegative Matrix Approximation problem, since factorization refers to an exact decomposition of the target matrix. However, the NMF is so ubiquitous that it stands for an approximation problem by convention.

1.2 Background

In the context of NMF, the given matrix X contains original nonnegative data and each column is a m-dimensional data sample. W is a matrix of basis vectors, where each column is a basis vector. Columns of W are not required to be orthogonal to each other. H is a weight matrix, in which each row is the gain of the corresponding basis vector. Therefore, $X = WH$ can be geometrically interpreted as every data points in X is contained in the cone generated by columns of W, $cone(W)$.

There are many challenging issues regarding the NMF. First, it is NP-hard [VA], which means it is very difficult to find an optimal solution. Unlike the unconstrained factorization problem which can be solved exactly with singular value decomposition, it is hard to find the global minimum of the target function, $f(W, H)$. Therefore, people have proposed many iterative algorithms that converge to stationary points. There exist many local minima due to the nonconvexity of $f(W, H)$ in both W and H. Therefore, NMF is not only an NP-hard problem, but also a problem for which we may not get an exact answer. The extreme difficulty has led to many proposed algorithms that are not easily compared theoretically. One interesting observation is that $f(W, H)$ is convex in either H or W. In other words, given a fixed H, we can efficiently find a unique solution W that minimizes $f(W, H)$ by using least square computation and for H given W. This observation is important, because it is used in one of the most efficient algorithms, Alternating Least Squares, to solve NMF. We discuss this in the following chapter.

The second challenge is that the NMF problem is ill-posed, because there does not exist a unique solution, even when the factorization is exact. For example, if WH is a solution, we can also find a pair of $W' = WD$ and $H' = D^{-1}H$ so that $W'H' = (WD)(D^{-1}H) = W(DD^{-1})H = WH$, where D is a positive diagonal matrix. We can easily check that $W' \in \mathbb{R}_+^{m\times r}$ and $H' \in \mathbb{R}_+^{r\times n}$. Geometrically, if we find a conic hull formed by columns of W that contains all data points in X, we can find a different conic hull formed by columns of another W', with a different weight matrix, which also contains all data points in X.

Despite such challenges, the NMF continues to be an area of much investigation, because of a growing variety of important applications, like image feature extraction [Le Se2] [Gu] and text mining [Ga] [Di Li].

1.3 Preliminary

1.3.1 Notations

$\mathbb{R}^{m\times n}$	set of $m \times n$ real matrices
$\mathbb{R}^{m\times n}$	set of $m \times n$ nonnegative real matrices
$\|\| \cdot \|\|$	Frobenius norm
$X_{i:}$	i^{th} row of matrix X
$X_{:j}$	j^{th} column of matrix X
$X_{:S}$	columns of matrix X indexed by the elements in set S
$X_{i:j}$	element located at the i^{th} row and the j^{th} column of matrix X
$cone(X)$	cone generated by columns of matrix X
$[X]_+$	projection of matrix X onto the nonnegative orthant
W^i	matrix W at the i^{th} iteration in an algorithm

$\sigma_1(X)$	maximal singular value of matrix X
$\arg\max_x f(x)$	value x that maximizes the value of the function f

1.3.2 Matrix Theory

Theorem 1.3.1. *(Singular Value Decomposition [Jo Ho]) Every matrix X with rank k can be represented as $X = \sum_{i=1}^{k} \sigma_i u_i v_i^T$, where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k > 0$ are positive singular values of X and $\{u_i, v_i\}_{i=1}^{k}$ are corresponding left and right singular vector pairs .*

Theorem 1.3.2. *(Eckart-Young-Mirsky Matrix Approximation Theorem [Go]) Given a matrix $A \in \mathbb{R}^{n \times m}$, the best rank r approximation to A is the sum of first r summands that is $X_r = \sum_{i=1}^{r} \sigma_i u_i v_i^T = \sum_{i=1}^{r} \sigma_i C_i = U_r \Sigma_r V_r^T$, in which X_r is a rank-r approximation, $C_i = u_i v_i^T$, columns of $U_r \in \mathbb{R}^{m \times r}$ (resp. of $V_r \in \mathbb{R}^{n \times r}$) are the left (resp. right) singular vectors, and Σ_r is a diagonal matrix containing singular values on its diagonal.*

1.4 Rank-one approximation

Lemma 1.4.1. *(Perron-Frobenius Theorem [Jo Ho]) Suppose $A \in \mathbb{R}^{n \times n}$ is nonnegative.*

There is an eigenvalue $\rho(A)$ that is real and positive with left and right eigenvectors.

For any other eigenvalue λ, $\rho(A) > |\lambda|$.

Since X is nonnegative, XX^T and $X^T X$ are nonnegative. Therefore, there are nonnegative left and right singular vectors u_1 and v_1 associated with the first singular value σ_1. According to Theorem 1.3.2, $\sigma_1 u_1 v_1^T$ is the optimal rank-one approximation of X. Since u_1 and v_1 are nonnegative, WH is the optimal nonnegative matrix factorization of X, given $r = 1$, in which $W = \sigma_1 u_1$ and $H = v_1^T$.

CHAPTER 2

EXISTING ALGORITHMS

In this chapter, I am going to describe existing algorithms from three main categories, which are multiplicative update methods, gradient descent methods, and alternating least squares (ALS) methods. ALS was first introduced by [Pa, Ta] for the positive matrix factorization. However, this problem did not gain much attention until [Le, Se] reintroduced it as NMF and proposed multiplicative update methods. This simple algorithm stimulated the wide research and applications of NMF not only in mathematics but also on image processing, text processing, bioinformatics, and etc. Another common practice to approach NP-hard problems is to use gradient descent. In each iteration, we need to calculate the gradient of our target function, $f(W, H)$, choose a suitable step size, and update matrices by taking a step in the direction of the negative gradient. Later, HALS, an improved version of ALS was introduced, which will converge more efficiently.

All of the above algorithms can be fitted into a general framework to solve NMF:

(a) Initialize starting matrices W^0 and H^0.

While do not satisfy stopping condition do:

(b) Fix H^i and update W^{i+1} such that $\|X - W^i H^i\|_F^2 \geq \|X - W^{i+1} H^i\|_F^2$.

(c) Fix W^{i+1} and update H^{i+1} such that $\|X - W^{i+1} H^i\|_F^2 \geq \|X - W^{i+1} H^{i+1}\|_F^2$.

The efficiency of each algorithm depends on how many computations are needed in (b) and (c) and how many iterations are needed for the algorithm to converge.

In the following sections, I will start by talking about multiplicative update method. Then, I will explain gradient descent and alternating least squares methods. In the last section, I will talk about singular value decomposition update method and why it fails.

2.1 Multiplicative Update

Algorithm 1: Multiplicative Update Method

```
initialize $W^0$ and $H^0$. $i = 0$ ;
while do not satisfy stopping condition do
    update $W^{i+1} = W^i \circ \frac{[X(H^i)^T]}{[W^i H^i (H^i)^T + \epsilon]}$ ;
    update $H^{i+1} = H^i \circ \frac{[(W^{i+1})^T X]}{[(W^{i+1})^T W^{i+1} H^i + \epsilon]}$ ;
    $i = i + 1$
end
```

Note that $\circ$ and $\frac{[.]}{[.]}$ denote the component-wise product and division. ϵ is a sufficient small but positive number to prevent the denominator becoming zero.

[Le Se] has shown that the Frobenius norm is non-increasing under the update rules, that is

$$||X - W^i H^i||_F^2 \geq ||X - W^{i+1} H^i||_F^2, \ ||X - W^{i+1} H^i||_F^2 \geq ||X - W^{i+1} H^{i+1}||_F^2.$$

[Le Se] also claimed that Algorithm 1 will converge to a local minimum, which was questioned by [Go, Zh] and [Li]. Moreover, when some entries in W and H become zero, those entries cannot be modified anymore and stay zero.

Therefore, [Be, Br] concludes that when Algorithm 1 converges to a limit point in the interior of the feasible region, the limit point is a stationary point, which, however, might be a saddle point. When Algorithm 1 converges to a limit point on the boundary of the feasible region, the stationary point cannot be determined.

Even though Algorithm 1 often converges in practice, it is slow to converge, especially when X is dense [Ha]. Since Algorithm 1 is a wildly used NMF algorithm, it is considered a baseline algorithm.

2.2 Projected Gradient Descent

Algorithm 2: Gradient Descent Method

```
initialize $W^0$ and $H^0$. $i = 0$ ;
while do not satisfy stopping condition do
    update $W^{i+1} = [W^i - \epsilon_{W^i}(W^i H^i - X)(H^i)^T]_+$ ;
    update $H^{i+1} = [H^i - \epsilon_{H^i}(W^{i+1})^T(W^{i+1}H^i - X)]_+$ ;
    $i = i + 1$
end
```

To minimize the objective function, $f(W, H)$ with nonnegative constraint, we need to use gradient descent with a projection function, $P(x)$ that maps x to the nearest feasible region. Here, we choose $P(x) = [x]_+$. In order to minimize $f(W, H)$, we need to find the stationary point by calculating the gradient.

$$\frac{\partial \|X - WH\|_F^2}{\partial W} = (WH - X)H^T$$

$$\frac{\partial \|X - WH\|_F^2}{\partial H} = W^T(WH - X)$$

Let ϵ_W and ϵ_H be step sizes. Then, we update W and H by

$$W^{new} = [W - \epsilon_W (WH - X)H^T]_+$$

$$H^{new} = [H - \epsilon_H W^T(WH - X)]_+$$

The convergence of Algorithm 2 depends on the choice of the step size. A poor choice of the step size like setting it to be a fraction might lead Algorithm 2 converge to a factorization not far from the starting matrices. [Li], [Jo], and [Da] have found smart choices for the step size and proved the convergence of their algorithm.

Algorithm 3 is also sensitive to the starting matrices. [Li] indicated that if W^0 and H^0 are starting matrices such that $\|X - W^0H^0\|_F^2 \geq \|X\|_F^2$, very often after the first iteration $W^1 = 0$ and $H^2 = 0$, making the algorithm stop. Therefore, a careful choice of starting matrices are needed.

2.3 Alternating Least Squares

Algorithm 3: Alternating Least Squares Method

```
initialize $W^0$ and $H^0$. $i = 0$ ;
while do not satisfy stopping condition do
    (1) Fix $H^i$ and solve for $W^{i+1}$: $W^{i+1} = \arg\min_{W \geq 0} \frac{1}{2} \|X - WH^i\|_F^2$ ;
    (2) Fix $W^{i+1}$ and solve for $H^{i+1}$: $H^{i+1} = \arg\min_{H \geq 0} \frac{1}{2} \|X - W^{i+1}H\|_F^2$ ;
    $i = i + 1$ ;
end
```

As discussed in the first chapter, the NMF is not convex in both W and H, but it is convex in either of W and W, when the other is fixed.

Based on this observation, NMF can be approached by iteratively solving a least squares problem with a nonnegative constraint. We consider (1) or (2) as a subproblem in Algorithm 3. Each subproblem can be decoupled into a collection of multiple nonnegative least squares problems. Take (2) as an example. We can solve (2) by solving each column of H^{i+1} from:

$$H^{i+1}_{:j} = min_{h \geq 0} \|X_{:j} - W^{i+1}h\|, \qquad (3)$$

where $H^{i+1}_{:j}$ is the jth column of H^{i+1} and $X_{:j}$ is the jth column of X. Methods in [La] and [Br] can be applied to solve the collection of nonnegative least squares problems. Algorithm 3 will converge to a local minimum, proved by [Gr, Sc] and [Li]. However, we need to solve $m + n$ nonnegative least squares problems in (1) and (2) per iteration, since W has m rows and H has n columns.

Hence, Algorithm 3 might be slower than Algorithm 1.

```
Algorithm 4: Practical Alternating Least Squares Method
initialize $W^0$ and $H^0$. $i = 0$;
while do not satisfy stopping condition do
    Fix $H$ and solve $W$: $WHH^T = XH^T$ ;
    $W = [W]_+$ ;
    Fix $W$ and solve $H$: $H^T W^T W = X^T W$ ;
    $H = [H]_+$ ;
    $i = i + 1$ ;
end
```

[La] proposed Algorithm 4 to address the issue of the high cost of time in Algorithm 3. It uses a standard unconstrained least squares method in [Bj] to solve (1) and (2) by ignoring the nonnegative constraint and projects the answer to the nonnegative orthant. There is no proof that Algorithm 4 will converge to a local minimum, so it may generate a saddle point. Therefore, Algorith 4 sacrifices the convergence property for speed.

To preserve the convergence property and speed up the algorithm, [Ci] designed an algorithm called hierarchical alternating least squares (HALS). In stead of solving a whole matrix at a time, HALS find H or W by successively updating a column in W and a row in H. In a single step of updating a column-row pair, we fix all other variables and the problem is reduced to

$$\min_{W_{:j},H_{j:}\geq 0} ||X - WH||_F^2 = \min_{W_{:j},H_{j:}\geq 0} ||(X - \sum_{k\neq j} W_{:k}H_{k:}) - W_{:j}H_{j:}||_F^2 = \min_{W_{:j},H_{j:}\geq 0} ||X_{(j)} - W_{:j}H_{j:}||_F^2, \quad (4)$$

where $X_{(j)} = X - \sum_{k\neq j} W_{:k}H_{k:}$ denotes the residue. In order to minimize (4), we need to find the stationary point by calculating the gradient.

$$\frac{\partial \|X_{(j)} - W_{:j}H_{j:}\|_F^2}{\partial W_{:j}} = W_{:j}H_{j:}H_{j:}^T - X_{(j)}H_{j:}^T = 0$$

$$\frac{\partial \|X_{(j)} - W_{:j}H_{j:}\|_F^2}{\partial H_{j:}} = W_{:j}^T W_{:j}H_{j:} - W_{:j}^T X_{(j)} = 0$$

Then, update $W_{:j}$ and $H_{j:}$ by

$$W_{:j}^{new} = \frac{[X_{(j)}H_{j:}^T]_+}{H_{j:}H_{j:}^T}$$

$$H_{j:}^{new} = \frac{[W_{:j}^T X_{(j)}]_+}{W_{:j}^T W_{:j}}$$

In practice, we may normalize the column vector $W_{:j}$ and the row vector $H_{j:}$ to unit length vectors after each update. Therefore, $W_{:j}^{new} = [X_{(j)}H_{j:}^T]_+$ and $H_{j:}^{new} = [W_{:j}^T X_{(j)}]_+$. We get the final update rules by substituting $X_{(j)} = X - \sum_{k \neq j} W_{:k}H_{k:} = X - WH + W_{:j}H_{j:}$.

$$W_{:j}^{new} = [(XH^T)_{:j} - W(HH^T)_{:j} + W_{:j}H_{j:}H_{j:}^T]_+$$

$$H_{j:}^{new} = [(W^T X)_{j:} - (W^T W)_{j:}H + W_{:j}^T W_{:j}H_{j:}]_+$$

Algorithm 5: Hierarchical Alternating Least Squares Method

```
initialize $W^0$ and $H^0$. $i = 0$. ;
while do not satisfy stopping condition do
    $A^i = X(H^i)^T$;
    $B^i = H^i(H^i)^T$;
    for $j = 1, 2, ..., r$ do
        $W^{i+1}_{:j} = [A^i_{:j} - W^i B^i_{:j} + W^i_{:j} H^i_{j:} (H^i)^T_{j:}]_+$ ;
    end
    $C^i = W^{i+1} X$;
    $D^i = (W^{i+1})^T W^{i+1}$;
    for $j = 1, 2, ..., r$ do
        $H^{i+1} = [C^i_{j:} - D^i_{j:} H + (W^{i+1})^T_{:j} W^{i+1}_{:j} H^i_{j:}]_+$ ;
    end
    $i = i + 1$ ;
end
```

Note that $W_{:j}$ and $H_{j:}$ only affect each other. In one iteration, only r columns in W and r rows in H. are updated Also note that XH^T and HH^T does not change when we are updating columns in W. Therefore, we calculate XH^T and HH^T before we update columns in W. Similarly, $W^T X$ and $W^T W$ does not change when we are updating rows in H. Thus, we calculate $W^T X$ and $W^T W$ before we update rows in H.

If we directly truncate negative elements to 0, we may have some zero blocks which will stay zero and cannot update in every iteration. Therefore, the algorithm cannot converge in this case. To solve this problem, we use $[]_+$ symbol, where $[x]_+ = max(\epsilon, x)$ and ϵ is a sufficiently small but positive value. You can check the convergence property of HALS in [Ho].

CHAPTER 3

NEAR SEPARABLE NMF

3.1 Problem statement

While NMF is NP-hard, [Ar] proved that NMF can be solved in polynomial time under the separability assumption. $X \in \mathbb{R}^{m\times n}$ is *r-separable* if $X = WH$, where $W \in \mathbb{R}^{m\times r}$, $H \in \mathbb{R}^{r\times n}$ and columns of W are a subset of columns of X. Geometrically, X is r-separable, if and only if all columns of X reside in the conical hull generated by a subset of r columns in X. Therefore, $X = WH$ can be rewritten as $X = X_{:S}H$, where S denotes the subset of r columns of X, whose conical hull contains all columns of X. [Ar] refers to these columns as anchors or extreme rays. If S is determined, then H can be easily calculated by solving a set of nonnegative least squares problems:

$$H_{:j} = \min_{h\geq 0} \|X_{:j} - X_{:S}h\|_F^2, \quad for\ j = 1, 2, ..., n$$

Note that H can be expressed as $[I_r,\ H^{'}]P$, where $I_r \in \mathbb{R}^{r\times r}$ identity matrix, $P \in \mathbb{R}^{n\times n}$ is a permutation matrix, and $H^{'} \in \mathbb{R}^{r\times(n-r)}$. Since H can be easily found given a fixed W, the problem can be reduced to find S which contains the anchors.

In real applications our target matrix X will not have an exact NMF with a lower inner dimension, so X will not be perfectly r-separable. Therefore, we want to develop algorithms to find an NMF of an r-separable matrix with some noise. A matrix X is *noisy r-separable* or *near separable*, if

$$X = X_{:S}H + N,$$

where $N \in \mathbb{R}^{m\times n}$ is a noise matrix with $\|N_{:i}\|_2 \leq \epsilon$ for all i for some sufficiently small ϵ. Geometrically, all columns of X approximately reside in the conical hull of its anchors. In the following section, two algorithms that are robust to noise are introduced.

3.2 Algorithms

There are two types of geometric approaches to solve the near-separable NMF. The first deals with convex hulls and the second deals with conical hulls. We will specifically discuss one representative algorithm for each approach in the following parts.

3.2.1 SPA

Lemma 3.2.1. *If X is r-separable and $D \in \mathbb{R}^{n\times n}$ is an invertible diagonal matrix, XD^{-1} is also r-separable.*

Proof. $XD^{-1} = X_{:S}HD^{-1} = X_{:S}D_{S:S}^{-1}D_{S:S}HD^{-1} = (XD^{-1})_{:S}H'$, where $H' = D_{S:S}HD^{-1} \in \mathbb{R}^{r\times n}$. □

Let D be a diagonal matrix, where $D_{i:i} = \|X_{:i}\|_1$. Then, XD^{-1} is separable by Lemma 3.2.1. Then the columns of XD^{-1} is normalized, while the entries of every column of H' sum to one. Then, every column in XD^{-1} can be as a data point. So, all points of XD^{-1} reside in the convex hull of r points in the set S. Then, the problem is reduce to finding the extreme points of a convex hull to find S. [Gi] applies Successive Projection Algorithm (SPA) proposed by [Ar Sa] to find the extreme points.

Algorithm 6: Successive Projection Algorithm

```
Set $R^1 = X$, $S^1 = \{\}$, and $i = 1$ ;
while $R^i \neq 0$ and $i \leq r$ do
    Solve for $j$: $j = \arg\max_j ||R^i_{:j}||_2^2$;
    $R^{i+1} = (I - \frac{R^i_{:j}(R^i_{:j})^T}{||R^i_{:j}||_2^2})R^i$;
    $S^{i+1} = S^i \cup \{j\}$ ;
    $i = i + 1$ ;
end
In the case of tie, the index $j$ whose corresponding column of the
  original matrix $X$ with maximum $l_2$ norm will be selected. If there is
  another tie, randomly select $j$ among those columns.
```

SPA works as follows: in the beginning, it lets the residual matrix be the target matrix. Then, it selects the point which has the greatest l_2 norm in the residual matrix, as it corresponds to an extreme vertex in the convex hull, which will be proved in the following part. Next, all data points are projected onto the orthogonal complement of the selected point. It repeats the process until r extreme points are selected. Note that after the first iteration, the residual matrix will typically have negative entries. However, it does not undermine the algorithm, since it does not set a nonnegative constraint to the residual matrix and it selects columns from the target matrix.

Why SPA can extract the extreme vertices is proved as follows [Gi].

Lemma 3.2.2. *Let $M = [W, 0^{m\times(r-k)}] \in \mathbb{R}^{m\times r}$, where $W \in \mathbb{R}^{m\times k}$ is full-rank and $r > k \geq 0$. Then,*

$$||Yh||_2^2 < \max_j ||W_{:j}||_2^2,\ \forall h \in \mathbb{R}^r \textit{ such that } h \neq e_i \textit{ for all } i$$

Proof. If $Yh = 0$, then $||Yh||_2^2 = 0$. Since W has full rank, $W_{:j} \neq 0$ for all j. Therefore, $||W_{:j}||_2^2 > 0 = ||Yh||_2^2$.

Therefore, assume $Yh = \sum_{j=1}^{k} h_j W_{:j}$, where $h_j \neq 0$ for at least one $1 \leq j \leq k$. Then,

$$||Yh||_F^2 = ||\sum_{j=1}^{k} h_j W_{:j}||_2^2 < \sum_{j=1}^{k} h_j ||W_{:j}||_2^2 \leq \max_j ||W_{:j}||_2^2$$

The first inequality is strict because $h \neq e_i$ for all i and h has at least one nonzero entry h_j. The second inequality is due to our construction that $\sum_i^k h_i \leq \sum_i^r h_i \leq 1$. □

Note that Lemma 3.2.2 implies that the column with the maximum l_2 norm in the matrix $X = WH$ will always be a column in W.

Theorem 3.2.3. *Let the matrix $X = WH$ with every column normalized. Then SPA recovers the set S such that $X_{:S} = W$ up to permutation.*

Proof. Let us prove this theorem by induction.

Base step: Since W has full rank, Lemma 3.2.2 applies. Therefore, the first iteration of SPA extracts a column in W. Assume without loss of generality that the column extracted is $W_{:j}$. Then, we can get the first residual matrix $R^1 = (I - \frac{W_{:j}W_{:j}^T}{||W_{:j}||_2^2})WH = [W^1, 0^{m\times 1}]H$. W^1 is full rank because W is.

Induction step: Assume that after k iterations the residual matrix is $R^k = [W^k, 0^{m\times k}]H$ with W^k full-rank. Then, the next iteration will extract a column that corresponds to one of the columns W^k. The next residual matrix will become $R^{k+1} = [W^{k+1}, 0^{m\times(k+1)}]$ where W^{k+1} has full rank since W^k has full rank. Therefore, after r iterations, SPA will extract all columns of W and the residual is zero. □

Thus, SPA can extract extreme vertices in the target matrix in the noiseless case if these extreme vertices are linearly independent. [Gi] also shows that SPA is robust when noise is present in our target matrix. However, Theorem 3.2.3 is based on the assumption that W has full rank. If the assumption is not met, SPA will fail to recover more than $rank(W)$ columns from W when W is not full rank, even if X is noiseless. In order to overcome this drawback, [Gi2] developed Successive Nonnegative Projection Algorithm by modifying the update rule for the residual matrix. SNPA is more computationally expensive but more robust than SPA.

3.2.2 X_{RAY}

X_{RAY} identifies an anchor by completing selection and projection steps in an iteration.

In the projection step, all data points are projected onto the current cone by solving the nonnegative least squares problem, $\arg\min_{H\geq 0} ||X - X_{:S^{i+1}}H||_F^2$ and compute a new residual matrix by $R = X - X_{:S^{i+1}}H$. Note that every residual column $R_{:k}$ is orthogonal to one of faces of the current cone after the projection step.

In the selection step, a face of the current cone is selected by picking a residual column $R_{:k}$. [Ku] observed and proved that for a given residual column $R_{:k}$ and any data $X_{:j}$ projected onto the current cone $cone(X_{:S^i})$, $R_{:k}^T X_{:j} \leq 0$ if $X_{:j}$ is inside the cone and $R_{:k}^T X_{:j} > 0$ if $X_{:j}$ is outside the cone. Therefore, all points that are exterior to the current cone are on one side of the hyperplane $R_{:k}^T x > 0$, while points contained in the cone are on the other side. The point $X_{:j}$ that maximizes the inner product $\frac{R_{:k}^T X_{:j}}{\|X_{:j}\|_1}$ is furthest from the hyperplane, so we select it as an anchor to expand the current cone. [Ku] defines multiple approaches (*rand*, *max*, and *dist*) to select a face of the current cone. All of these approaches can find anchors in separable cases but behave differently in the presence of noise.

Algorithm 7: X_{RAY}

```
Set $R = X$, $S^i = \{\}$, and $i = 1$ ;
while $R \neq 0$ and $i \leq r$ do
    Select $k$ according to one of following criteria:
        $rand$ : any random $k$ such that $\|R_{:k}\|_2$ ;
        $max$ : solve for $k$: $\max_k \|R_k\|$ ;
        $dist$ : solve for $k$: $\max_k \|[R_k^T X]_+\|_2$ ;
    Solve for j: $j = \arg\max_j \frac{R_{:k}^T X_{:j}}{\|X_{:j}\|_1}$
    $S^{i+1} = S^i \cup \{j\}$ ;
    Solve for $H$: $H = \arg\min_{H \geq 0} \|X - X_{:S^{i+1}} H\|_F^2$
    $R = X - X_{:S^{i+1}} H$
end
```

Although X_{RAY} only takes r iterations to complete, it is not computationally cheap because it has to solve the nonnegative least squares problem in the projection step. Furthermore, even without the presence of noise, X_{RAY} will fail to extract a column of W if there are more than two columns that maximizes the inner product and one of these columns is a conic combination of the others. [Ku] did not give a rigorous analysis of X_{RAY} in the near-separable case, which makes this algorithm mostly empirical.

CHAPTER 4

INITIALIZATION

As mentioned in chapter 1, NMF is NP-hard, so the best one can do is to use iterative methods to find local minima. Starting too far from a stationary point, too many iterations will be needed to converge. Therefore, good initialization strategies are needed to converge to a competitive stationary point and reduce convergence time.

4.1 The SVD

The singular value decomposition (SVD) is a standard tool to generate a lower-rank matrix decomposition when there are no constraints. According to Eckart-Young-Mirsky Matrix Approximation Theorem, the best rank r approximation to X is the sum of first r summands that is $X_r = \sum_{i=1}^{r} \sigma_i u_i v_i^T = \sum_{i=1}^{r} \sigma_i C_i = U_r \Sigma_r V_r^T$, in which X_r is a rank-r approximation, $C_i = u_i v_i^T$, columns of $U_r \in \mathbb{R}^{m \times r}$ (resp. of $V_r \in \mathbb{R}^{n \times r}$) are the left (resp. right) singular vectors, and Σ_r is a diagonal matrix containing singular values on its diagonal. However, with the nonnegativity constraint, we can only guarantee that C_1 is nonnegative by Lemma 1.4.1, while for $i > 1$, C_i typically has negative entries. Therefore, U_r and $\Sigma_r V_r$ cannot be used directly as a starting point. There are two methods that have been used to modify the SVD to get a nonnegative approximation.

[Qi] proposed an algorithm called SVD-NMF which approximates X by $|U_r| \Sigma_r |V_r^T|$. It initializes $W = |U_r|$ and $H = \Sigma_r |V_r^T|$. However, simply taking the absolute value will lose the sign information. There is no theoretical analysis for error bounds and performance of this approach in the existing literature.

[Bo] proposed a more widely used algorithm called the nonnegative double SVD (NNDSVD). It uses the SVD on the positive part of C_i. It further decomposes X_r as follows:

$$X_r = \sum_{i=1}^{r} \sigma_i C_i = \sigma_1 C_1 + \sum_{i=2}^{r} \sigma_i C_i = \sigma_1 C_1 + \sum_{i=2}^{r} \sigma_i C_i^+ - \sum_{i=2}^{r} \sigma_i C_i^-$$

$$= \sigma_1 C_1 + \sum_{i=2}^{r} \sigma_i \sigma_1(C_i^+) u_1(C_i^+) v_1(C_i^+)^T - \sum_{i=2}^{r} \sigma_i \sigma_2(C_i^+) u_2(C_i^+) v_2(C_i^+)^T \sum_{i=2}^{r} -\sigma_i C_i^-,$$

in which $\sigma_i(C_i^+)$ is the ith largest singular value of C_i^+ and $u_i(C_i^+)$ and $v_i(C_i^+)$ are the ith left and right singular vectors of C_i^+. Note that [Bo] proved that $rank(C_i^+) \leq 2$, so C_i has at most two singular values. Since C_i^+ is nonnegative, its rank-1 approximation is also nonnegative by the Perron-Frobenius theorem.

NNDSVD approximates X_r by truncating the negative part. For the starting matrices W and H, the first column (resp. row) of W (resp. H) will be $\sigma_1^{1/2} u_1$ (resp. $\sigma_1^{1/2} v_1^T$) and the ith column (resp. row) of W (resp. H) will be $(\sigma_i \sigma_1(C_i^+))^{1/2} u_1(C_i^+)$ (resp. $(\sigma_i \sigma_1(C_i^+))^{1/2} v_1(C_i^+)^T$).

Note that for r sufficiently large, the difference between X and its rank-r initial approximation will grow as r grows, since more negative entries are truncated. This is a problem in both initialization strategies since it would make more sense that the approximation will be better as r grows, as in the unconstrained rank-r approximation. [Sy] modified the initialization strategy to overcome this problem.

4.2 Clustering

[Di] showed that the K-means clustering on nonnegative data is equivalent to NMF. [Wi] directly applied the K-means clustering method proposed by [Ma] to initialize columns in W, taking the columns of X as data to be clustered and columns of W as centroids. H is initialized as the cluster indicator matrix ($H_{i,j} = 1$ if $X_{:j}$ belongs to the ith cluster). Since the K-means clustering method is itself iterative, the main drawback of this initialization strategy is its expensive cost.

4.3 SPA

As discussed in Chapter 3, SPA is an efficient and reproducible method to find extreme points in a convex hull in the separable case. However, without the separability assumption, the extreme points may not present in the points in target matrix. Therefore, SPA cannot be directly used for the non-separable case. However, it might provide a good starting point, since the key idea of SPA is to find points such that the geometrical hull formed by these points is as large as possible. SPA gives a more structured starting point than the random initialization and is faster than SVD-based or clustering initializations. Even if SPA does not select best points, the NMF algorithm can still adjust the approximation. However, SPA is sensitive to outliers. For example, if we have a perfect convex hull that contains all points except one outlier, SPA might select this outlier, since it wants to expand the hull.

CHAPTER 5

EXPERIMENTS AND CONCLUSION

5.1 Experiments

In this section, four NMF algorithms and three Initialization strategies are tested on the CBCL data set [CB] . CBCL data set contains 2429 greyscale 19×19 facial images. Therefore, X, the matrix obtained from this data set is a 361×2429 nonnegative matrix, in which each column is a vetorized representation of an image. 49 basis images are used to approximate X, so $r = 49$.

The first experiment was conducted by running MU, CD, ALS, and HALS on X, with the same randomly initialized matrices twenty times. The average error was obtained by taking the average of each time. The second (third) experiment were conducted by running MU (HALS) with SVD, SPA, and random initializations.

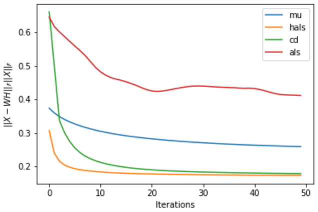

Figure 5.1: Errors by using algorithms MU, CD, ALS, HALS

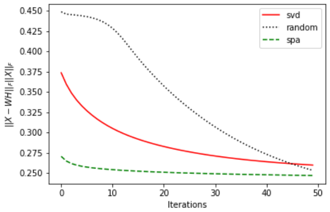

Figure 5.2: Errors by using MU, initialized with SPA, SVD, random

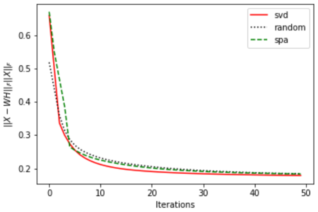

Figure 5.3: Errors by using HALS, initialized with SPA, SVD, random

From Figure 5.1, we can observe:

- ALS oscillates and does not converge.

- MU converges relatively slowly
- HALS converges the fastest and the error is the lowest
- The error of CD is comparable to HALS.

From Figure 5.2 and Figure 5.3, we can observe:

- For the MU algorithm, SPA will give the best starting matrices and converge the fastest.
- For the HALS algorithm, three initialization strategies are comparable and the SVD method is slightly better.

5.2 Conclusion

The NMF is a useful dimensionality reduction technique for nonnegative data, with wide applications in different disciplines. Due to its usefulness and extreme difficulty, the NMF has led to abundant research. Although there may not be an optimal solution for the general NMF, researchers have solved some extensions of the NMF that have some nice properties, like the nonnegative factorization of a separable matrix. With the greater computational capacity of current computers, more heuristic algorithms have emerged. Meanwhile, we believe more fundamental theories underlying this problem also need to be explored and studied.

BIBLIOGRAPHY

[Te Pa] Paatero, Pentti and Unto Tapper. “Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values.” (1994).

[Le Se] Lee, Daniel Seung, Hyunjune. (2001). Algorithms for Non-negative Matrix Factorization. Adv. Neural Inform. Process. Syst.. 13.

[VA] Vavasis, Stephen. (2007). On the Complexity of Nonnegative Matrix Factorization. SIAM Journal on Optimization. 20. 10.1137/070709967.

[Le Se2] Lee, Daniel Seung, H.. (1999). Learning the parts of objects by nonnegative matrix factorization. Nature. 401.

[Gu] Guillamet, David Vitrià, Jordi. (2002). Non-negative Matrix Factorization for Face Recognition. Topics in Artificial Intelligence. 35. 336-344. 10.1007/3-540-36079-4-29.

[Ga] Gaussier, Eric Goutte, Cyril. (2005). Relation between PLSA and NMF and implications. Proc. Annual Int. SIGIR Conf. Res. Develop. Inform. Retrieval (SIGIR’05). 28. 601-602. 10.1145/1076034.1076148.

[Di Li] Ding, Chris Li, Tao Peng, Wei, 2008. ”On the equivalence between Non-negative Matrix Factorization and Probabilistic Latent Semantic Indexing,” Computational Statistics Data Analysis, Elsevier, vol. 52(8), pages 3913-3927, April.

[Li] Lin, Chih-Jen. (2007). On the Convergence of Multiplicative Update Algorithms for Nonnegative Matrix Factorization. Neural Networks, IEEE Transactions on. 18. 1589 - 1596. 10.1109/TNN.2007.895831.

[Be, Br] Berry, Michael Browne, Murray Langville, Amy Pauca, V.Paul Plemmons, Robert. (2007). Algorithms and Applications for Approximate Nonnegative Matrix Factorization. Computational Statistics Data Analysis. 52. 155-173. 10.1016/j.csda.2006.11.006.

[Ha] Han, Jian Han, Lixing NEUMANN, M Prasad, Upendra. (2009). On the rate of convergence of the image space reconstruction algorithm. Operators and Matrices. 3. 10.7153/oam-03-02.

[Jo] Johansson, Björn Elfving, Tommy Kozlov, V. Censor, Yair Forssén, Per-Erik Granlund, Gösta. (2006). The application of an oblique-projected Landweber method to a model of supervised learning. Mathematical and Computer Modelling. 43. 892-909. 10.1016/j.mcm.2005.12.010.

[Da] Dai, Yu-Hong Fletcher, Roger. (2005). Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. Numerische Mathematik. 100. 21-47. 10.1007/s00211-004-0569-y.

[La] Lawson, C. Hanson, Richard. (2020). Solving least squares problems prentice-hall.

[Br] Bro, Rasmus Jong, Sijmen. (1997). A Fast Non-negativity-constrained Least Squares Algorithm. Journal of Chemometrics. 11. 393-401. 10.1002/(SICI)1099-128X(199709/10)11:53.0.CO;2-L.

[Li] Lin, Chih-Jen. (2007). Projected Gradient Methods for Non-Negative Matrix Factorization. Neural computation. 19. 2756-79. 10.1162/neco.2007.19.10.2756.

[Gr, Sc] Grippo, L.. (2000). On the convergence of the block nonlinear Gauss-Seidel method under convex constraints. Operations Research Letters. 26. 127-136. 10.1016/S0167-6377(99)00074-7.

[La] Albright, Russell Cox, James Duling, David Langville, Amy Meyer, Carl. (2014). Algorithms, initializations, and convergence for the nonnegative matrix factorization. Proc. 12th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining.

[Bj] Björck, ke,. (1996). Numerical Methods for Least Squares Problems. SIAM Philadelphia. 10.1137/1.9781611971484.

[Ho] Ho, Ngoc-Diep. (2008). Nonnegative matrix factorization algorithms and applications. PhD thesis.

[Ar] Arora, Sanjeev Ge, Rong Kannan, Ravi Moitra, Ankur. (2016). Computing a Nonnegative Matrix Factorization—Provably. SIAM Journal on Computing. 45. 1582-1611. 10.1137/130913869.

[Gi] Gillis, Nicolas Vavasis, Stephen. (2013). Fast and Robust Recursive Algorithms for Separable Nonnegative Matrix Factorization. IEEE transactions on pattern analysis and machine intelligence.

[Ar Sa] Bezerra, Saldanha Galvão, Roberto Yoneyama, Takashi Chame, Henrique Visani, Valeria. (2001). The successive projections algorithm for variable selection in spectroscopic multicomponent analysis. Chemometrics and Intelligent Laboratory Systems. 57. 65-73. 10.1016/S0169-7439(01)00119-8.

[Ku] Kumar, Abhishek Sindhwani, Vikas Kambadur, Prabhanjan. (2012). Fast Conical Hull Algorithms for Near-separable Non-negative Matrix Factorization.

[Go] Golub, G.H. Hoffman, Alan Stewart, G.W.. (1987). A generalization of the Eckart-Young-Mirsky matrix approximation theorem. Linear Algebra and its Applications. s 88–89. 317–327. 10.1016/0024-3795(87)90114-5.

[Qi] Qiao, Hanli. (2014). New SVD Based Initialization Strategy for Non-negative Matrix Factorization. Pattern Recognition Letters. 63. 10.1016/j.patrec.2015.05.019.

[Bo] Boutsidis, Christos Gallopoulos, Efstratios. (2008). Gallopoulos, E.: Svd based initialization: A head start for nonnegative matrix factorization. Pattern Recognition 41(4), 1350-1362. Pattern Recognition. 1350 – 1362. 1350-1362. 10.1016/j.patcog.2007.09.010. [Sy] Atif, Syed Qazi, Sameer Gillis, Nicolas. (2019). Improved SVD-based initialization for nonnegative matrix factorization using low-rank correction. Pattern Recognition Letters. 122. 10.1016/j.patrec.2019.02.018.

[Wi] Wild, Stefan Curry, James Dougherty, Anne. (2004). Improving nonnegative matrix factorization through structured initialization. Pattern Recognition. 37. 2217-2232. 10.1016/j.patcog.2004.02.013.

[Ma] McQueen, J.. (1967). Some methods for classification and analysis of multivariate observations. Computer and Chemistry. 4. 257-272.

[Di] Ding, Chris He, Xiaofeng Simon, Horst Jin, Rong. (2005). On the Equivalence of Nonnegative Matrix Factorization and K-means- Spectral Clustering. Proceedings of the 2005 SIAM International Conference on Data Mining. 10.1137/1.9781611972757.70.

[Jo Ho] Horn, R. A. Johnson, C. R. (1990). Matrix Analysis. Cambridge University Press.

[CB] MIT CBCL facial database, http://cbcl.mit.edu/cbcl/software-datasets/FaceData2.html