2

Nonnegative Matrix Factorization

In this chapter, we will explore the nonnegative matrix factorization problem. It will be helpful to first compare it to the more familiar singular value decomposition. In the worst case, the nonnegative matrix factorization problem is *NP*-hard (seriously, what else did you expect?), but we will make domain-specific assumptions (called *separability*) that will allow us to give provable algorithms for an important special case of it. We then apply our algorithms to the problem of learning the parameters of a topic model. This will be our first case study in how to not back down in the face of computational intractability, and to find ways around it.

2.1 Introduction

In order to better understand the motivations behind the nonnegative matrix factorization problem and why it is useful in applications, it will be helpful to first introduce the singular value decomposition and then compare the two. Eventually, we will apply both of these to text analysis later in this section.

The Singular Value Decomposition

The singular value decomposition (SVD) is one of the most useful tools in linear algebra. Given an $m \times n$ matrix M, its singular value decomposition is written as

$$M = U\Sigma V^T$$

where U and V are orthonormal and Σ is a rectangular matrix with nonzero entries only along the diagonal, and its entries are nonnegative. Alternatively, we can write

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

$$M = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

where u_i is the i^{th} column of U, v_i is the i^{th} column of V, and σ_i is the i^{th} diagonal entry of Σ. Throughout this section we will fix the convention that $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. In this case, the rank of M is precisely r.

Throughout this book, we will have occasion to use this decomposition as well as the (perhaps more familiar) eigendecomposition. If M is an $n \times n$ matrix and is diagonalizable, its eigendecomposition is written as

$$M = PDP^{-1}$$

where D is diagonal. For now, the important facts to remember are:

(1) **Existence:** Every matrix has a singular value decomposition, even if it is rectangular. In contrast, a matrix must be square to have an eigendecomposition. Even then, not all square matrices can be diagonalized, but a sufficient condition under which M can be diagonalized is that all its eigenvalues are distinct.
(2) **Algorithms:** Both of these decompositions can be computed efficiently. The best general algorithms for computing the singular value decomposition run in time $O(mn^2)$ if $m \geq n$. There are also faster algorithms for sparse matrices. There are algorithms to compute an eigendecomposition in $O(n^3)$ time and there are further improvements based on fast matrix multiplication, although it is not clear whether such algorithms are as stable and practical.
(3) **Uniqueness:** The singular value decomposition is unique if and only if its singular values are distinct. Similarly, the eigendecomposition is unique if and only if its eigenvalues are distinct. In some cases, we will only need that the nonzero singular values/eigenvalues are distinct, because we can ignore the others.

Two Applications

Two of the most important properties of the singular value decomposition are that it can be used to find the best rank k approximation and that it can be used for dimension reduction. We explore these next. First, let's formalize what we mean by the best rank k approximation problem. One way to do this is to work with the Frobenius norm:

Definition 2.1.1 (Frobenius norm) $\|M\|_F = \sqrt{\sum_{i,j} M_{i,j}^2}$

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

It is easy to see that the Frobenius norm is invariant under rotations. For example, this follows by considering each of the columns of M separately as a vector. The square of the Frobenius norm of a matrix is the sum of squares of the norms of its columns. Then left-multiplying by an orthogonal matrix preserves the norm of each of its columns. An identical argument holds for right-multiplying by an orthogonal matrix (but working with the rows instead). This invariance allows us to give an alternative characterization of the Frobenius norm that is quite useful:

$$\|M\|_F = \|U^T M V\|_F = \|\Sigma\|_F = \sqrt{\sum \sigma_i^2}$$

The first equality is where all the action is happening and uses the rotational invariance property we established above.

Then the Eckart–Young theorem asserts that the best rank k approximation to some matrix M (in terms of Frobenius norm) is given by its truncated singular value decomposition:

Theorem 2.1.2 (Eckart–Young) $\operatorname{argmin}_{\text{rank}(B)\leq k} \|M - B\|_F = \sum_{i=1}^{k} \sigma_i u_i v_i^T$

Let M_k be the best rank k approximation. Then, from our alternative definition of the Frobenius norm, it is immediate that $\|M - M_k\|_F = \sqrt{\sum_{i=k+1}^{r} \sigma_i^2}$.

In fact, the same statement – that the best rank k approximation to M is its truncated singular value decomposition – holds for *any* norm that is invariant under rotations. As another application, consider the operator norm:

Definition 2.1.3 (operator norm) $\|M\| = \max_{\|x\|\leq 1} \|Mx\|$

It is easy to see that the operator norm is also invariant under rotations and, moreover, $\|M\| = \sigma_1$, again using the convention that σ_1 is the largest singular value. Then the Eckart–Young theorem with respect to the operator norm asserts:

Theorem 2.1.4 (Eckart–Young) $\operatorname{argmin}_{\text{rank}(B)\leq k} \|M - B\| = \sum_{i=1}^{k} \sigma_i u_i v_i^T$

Again, let M_k be the best rank k approximation. Then $\|M - M_k\| = \sigma_{k+1}$. As a quick check, if $k \geq r$ then $\sigma_{k+1} = 0$, and the best rank k approximation is exact and has no error (as it should). You should think of this as something you can do with any algorithm to compute the singular value decomposition of M – you can find the best rank k approximation to it with respect to any rotationally invariant norm. In fact, it is remarkable that the best rank k approximation in many different norms coincides! Moreover, the best rank k approximation to

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

M can be obtained directly from its best rank $k+1$ approximation. This is not always the case, as we will see in the next chapter when we work with tensors.

Next, we give an entirely different application of the singular value decomposition in the context of data analysis before we move on to applications of it in text analysis. Recall that M is an $m \times n$ matrix. We can think of it as defining a distribution on n-dimensional vectors, which we obtain from choosing one of its columns uniformly at random. Further suppose that $\mathbb{E}[x] = 0$; i.e., the columns sum to the all-zero vector. Let $\mathcal{P}_k$ be the space of all projections onto a k-dimensional subspace.

Theorem 2.1.5 $\operatorname{argmax}_{P \in \mathcal{P}_k} \mathbb{E}[\|Px\|^2] = \sum_{i=1}^{k} u_i u_i^T$

This is another basic theorem about the singular value decomposition, and from it we can readily compute the k-dimensional projection that maximizes the projected variance. This theorem is often invoked in visualization, where one can visualize high-dimensional vector data by projecting it to a more manageable lower-dimensional subspace.

Latent Semantic Indexing

Now that we have developed some of the intuition behind the singular value decomposition, we will see an application of it to text analysis. One of the central problems in this area (and one that we will return to many times) is this: given a large collection of documents, we want to extract some hidden *thematic* structure. Deerwester et al. [60] invented latent semantic indexing (LSI) for this purpose, and their approach was to apply the singular value decomposition to what is usually called the term-by-document matrix:

Definition 2.1.6 *The term-by-document matrix M is an $m \times n$ matrix where each row represents a word and each column represents a document where*

$$M_{i,j} = \frac{\text{count of word } i \text{ in document } j}{\text{total number of words in document } j}.$$

There are many popular normalization conventions, and here we have chosen to normalize the matrix so that each of its columns sums to one. In this way, we can interpret each document as a probability distribution on words. Also, in constructing the term-by-document matrix, we have ignored the order in which the words occur. This is called a *bag-of-words representation*, and the justification for it comes from a thought experiment. Suppose I were to give you the words contained in a document, but in a jumbled order. It should still be possible to determine what the document is about, and hence forgetting all

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

notions of syntax and grammar and representing a document as a vector loses some structure, but should preserve enough of the information to make many basic tasks in text analysis still possible.

Once our data is in vector form, we can make use of tools from linear algebra. How can we measure the similarities between two documents? The naive approach is to base our similarity measure on how many words they have in common. Let's try

$$\langle M_i, M_j \rangle.$$

This quantity computes the probability that a randomly chosen word w from document i and a randomly chosen word w' from document j are the same. But what makes this a bad measure is that when documents are sparse, they may not have many words in common just by accident because of the particular words each author chose to use to describe the same types of things. Even worse, some documents could be deemed to be similar because they contain many of the same common words, which have little to do with what the documents are actually about.

Deerwester et al. [60] proposed to use the singular value decomposition of M to compute a more reasonable measure of similarity, and one that seems to work better when the term-by-document matrix is sparse (as it usually is). Let $M = U\Sigma V^T$ and let $U_{1\ldots k}$ and $V_{1\ldots k}$ be the first k columns of U and V, respectively. The approach is to compute

$$\langle U_{1\ldots k}^T M_i, U_{1\ldots k}^T M_j \rangle$$

for each pair of documents. The intuition is that there are some *topics* that occur over and over again in the collection of documents. And if we could represent each document M_i on the basis of topics, then their inner product on that basis would yield a more meaningful measure of similarity. There are some models – i.e., hypotheses for how the data is stochastically generated – where it can be shown that this approach provably recovers the true topics [118]. This is the ideal interaction between theory and practice – we have techniques that work (somewhat) well, and we can analyze/justify them.

However, there are many failings of latent semantic indexing that have motivated alternative approaches. If we associate the top singular vectors with topics, then

(1) topics are orthonormal.

However, topics like *politics* and *finance* actually contain many words in common, so they cannot be orthonormal.

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

(2) topics contain negative values.

Hence, if a document contains such words, their contribution (to the topic) could cancel out the contributions from other words. Moreover, a pair of documents can be judged to be similar because of particular topics that they are both not about.

Nonnegative Matrix Factorization

For exactly the failings we described in the previous section, nonnegative matrix factorization is a popular alternative to the singular value decomposition in many applications in text analysis. However, it has its own shortcomings. Unlike the singular value decomposition, it is *NP*-hard to compute. And the prevailing approach in practice is to rely on heuristics, with no provable guarantees.

Definition 2.1.7 *A nonnegative matrix factorization of inner-dimension r is a decomposition*

$$M = AW$$

where A is $n \times r$, W is $r \times n$, and both are entrywise nonnegative. Moreover, let the nonnegative rank of M – denoted by $\text{rank}^+(M)$ *– be the minimum r so that such a factorization exists.*

As we will see, this factorization, when applied to a term-by-document matrix, can find more interpretable topics. Beyond text analysis, it has many other applications in machine learning and statistics, including in collaborative filtering and image segmentation. For now, let's give an interpretation of a nonnegative matrix factorization specifically in the context of text analysis. Suppose we apply it to a term-by-document matrix. Then it turns out that we can always put it in a convenient canonical form: Let D be a diagonal matrix where

$$D_{j,j} = \sum_{i=1}^{m} A_{i,j}$$

and further suppose that each $D_{j,j} > 0$. Then

Claim 2.1.8 *Set $\widetilde{A} = AD^{-1}$ and $\widetilde{W} = DW$. Then*

(1) $\widetilde{A}, \widetilde{W}$ are entrywise nonnegative and $M = \widetilde{A}\widetilde{W}$, and
(2) the columns of $\widetilde{A}$ and the columns of $\widetilde{W}$ each sum to one.

We leave the proof of this claim as an exercise, but the hint is that property (2) follows because the columns of M also sum to one.

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Hence we can, without loss of generality, assume that our nonnegative matrix factorization $M = AW$ is such that the columns of A and the columns of W each sum to one. Then we can interpret this factorization as follows: Each document is itself a distribution on words, and what we have found is

(1) a collection of r topics – the columns of A – that are themselves distributions on words, and
(2) for each document i, a representation of it – given by W_i – as a convex combination of r topics so that we recover its original distribution on words.

Later on, we will get some insight into why nonnegative matrix factorization is *NP*-hard. But what approaches are used in practice to actually compute such a factorization? The usual approach is *alternating minimization*:

Alternating Minimization for NMF

Input: $M \in \mathbb{R}^{m \times n}$
Output: $M \approx A^{(N)} W^{(N)}$

Guess entrywise nonnegative $A^{(0)}$ of dimension $m \times r$
For $i = 1$ to N
    Set $W^{(i)} \leftarrow \text{argmin}_W \|M - A^{(i-1)} W\|_F^2$ s.t. $W \geq 0$
    Set $A^{(i)} \leftarrow \text{argmin}_A \|M - AW^{(i)}\|_F^2$ s.t. $A \geq 0$
End

Alternating minimization is quite general, and throughout this book we will come back to it many times and find that problems we are interested in are solved in practice using some variant of the basic approach above. However, it has no provable guarantees in the traditional sense. It can fail by getting stuck in a locally optimal solution that is much worse than the globally optimal one. In fact, this is inevitable, because the problem it is attempting to solve really is *NP*-hard.

However, in many settings we will be able to make progress by working with an appropriate stochastic model, where we will be able to show that it converges to a globally optimal solution provably. A major theme in this book is to not take for granted heuristics that seem to work in practice "as immutable," because the ability to analyze them will itself provide new insights into when and why they work, and also what can go wrong and how to improve them.

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

2.2 Algebraic Algorithms

In the previous section, we introduced the nonnegative matrix factorization problem and described some of its applications in machine learning and statistics. In fact, because of the algebraic nature of the problem, it is far from clear that there is any finite time algorithm for computing it in the worst case. Here we will explore some of the fundamental results in solving systems of polynomial equations, and derive algorithms for nonnegative matrix factorization from these.

Rank vs. Nonnegative Rank

Recall that $\text{rank}^+(M)$ is the smallest value r such that M has a nonnegative matrix factorization $M = AW$ with inner dimension r. It is easy to see that the following is an equivalent definition:

Claim 2.2.1 *$\text{rank}^+(M)$ is the smallest r such that there are r entrywise nonnegative rank one matrices $\{M_i\}$ that satisfy $M = \sum_i M_i$.*

We can now compare the rank and the nonnegative rank. There are, of course, many equivalent definitions for the rank of a matrix, but the most convenient definition to compare the two is the following:

Claim 2.2.2 *$\text{rank}(M)$ is the smallest r such that there are r rank one matrices $\{M_i\}$ that satisfy $M = \sum_i M_i$.*

The only difference between these two definitions is that the former stipulates that all of the rank one matrices in the decomposition are entrywise nonnegative, while the latter does not. Thus it follows immediately that

Fact 2.2.3 *$\text{rank}^+(M) \geq \text{rank}(M)$*

Can the nonnegative rank of a matrix be much larger than its rank? We encourage the reader to think about this question before proceeding. This is equivalent to asking whether, for an entrywise nonnegative matrix M, one can, without loss of generality, require the factors in its rank decomposition to be entrywise nonnegative too. It is certainly true for a rank one matrix, and turns out to be true for a rank two matrix too, but...

In general, the nonnegative rank cannot be bounded by any function of the rank alone. In fact, the relationship (or lack thereof) between the rank and the nonnegative rank is of fundamental importance in a number of areas in theoretical computer science. Fortunately, there are simple examples that illustrate that the two parameters can be far apart:

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Example: Let M be an $n \times n$ matrix where $M_{ij} = (i-j)^2$.

It is easy to see that the column space of M is spanned by the following three vectors:

$$\begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \\ \vdots \\ n \end{bmatrix}, \begin{bmatrix} 1 \\ 4 \\ \vdots \\ n^2 \end{bmatrix}.$$

Hence $\text{rank}(M) \leq 3$. (In fact, $\text{rank}(M) = 3$.) However, M has zeros along the diagonal and nonzeros off the diagonal. Furthermore, for any rank one entrywise nonnegative matrix M_i, its pattern of zeros and nonzeros is a *combinatorial rectangle* – i.e., the intersection of some set of rows and columns – and it can be shown that one needs at least $\log n$ such rectangles to cover the nonzeros of M without covering any of its zeros. Hence:

Fact 2.2.4 *$\text{rank}^+(M) \geq \log n$*

A word of caution: For this example, a number of authors have incorrectly tried to prove a much stronger lower bound (e.g., $\text{rank}^+(M) = n$). In fact (and somewhat surprisingly), it turns out that $\text{rank}^+(M) \leq 2 \log n$. The usual error is in thinking that because the rank of a matrix is the largest r such that it has r linearly independent columns, the nonnegative rank is the largest r such that there are r columns where no column is a convex combination of the other $r-1$. This is not true!

Systems of Polynomial Inequalities

We can reformulate the problem of deciding whether $\text{rank}^+(M) \leq r$ is a problem of finding a feasible solution to a particular system of polynomial inequalities. More specifically, $\text{rank}^+(M) \leq r$ if and only if

$$\begin{cases} M = AW \\ A \geq 0 \\ W \geq 0 \end{cases} \tag{2.1}$$

has a solution. This system consists of quadratic equality constraints (one for each entry of M) and linear inequalities that require A and W to be entrywise nonnegative. Before we worry about fast algorithms, we should ask a more basic question (whose answer is not at all obvious):

Question 4 *Is there any finite time algorithm for deciding if $\text{rank}^+(M) \leq r$?*

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

This is equivalent to deciding if the above linear system has a solution, but the difficulty is that even if there is one, the entries of A and W could be irrational. This is quite different than, say, 3-SAT, where there is a simple brute-force algorithm. In contrast, for nonnegative matrix factorization it is quite challenging to design algorithms that run in any finite amount of time.

But indeed there are algorithms (that run in some fixed amount of time) to decide whether a system of polynomial inequalities has a solution or not in the real RAM model. The first finite time algorithm for solving a system of polynomial inequalities follows from the seminal work of Tarski, and there has been a long line of improvements based on successively more powerful algebraic decompositions. This line of work culminated in the following algorithm of Renegar:

Theorem 2.2.5 *[126] Given a system of m polynomial inequalities in k variables, whose maximum degree is D and whose bit complexity is L, there is an algorithm whose running time is*

$$(nDL)^{O(k)}$$

that decides whether the system has a solution. Moreover, if it does have a solution, then it outputs a polynomial and an interval (one for each variable) in which there is only one root, which is the value of the variable in the true solution.

Notice that this algorithm finds an implicit representation of the solution, since you can find as many bits of the solution as you would like by performing a binary search for the root. Moreover, this algorithm is essentially optimal, and improving it would yield subexponential time algorithms for 3-SAT.

We can use these algorithms to solve nonnegative matrix factorization, and it immediately implies that there is an algorithm for deciding if $\text{rank}^+(M) \leq r$ runs in exponential time. However, the number of variables we would need in the naive representation is $nr + mr$, one for each entry in A or W. So even if $r = O(1)$, we would need a linear number of variables and the running time would still be exponential. It turns that even though the naive representation uses many variables, there is a more clever representation that uses many fewer variables.

Variable Reduction

Here we explore the idea of finding a system of polynomial equations that expresses the nonnegative matrix factorization problem using many fewer

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

variables. In [13, 112], Arora et al. and Moitra gave a system of polynomial inequalities with $f(r) = 2r^2$ variables that has a solution if and only if $\text{rank}^+(M) \leq r$. This immediately yields a polynomial time algorithm to compute a nonnegative matrix factorization of inner-dimension r (if it exists) for any $r = O(1)$. These algorithms turn out to be essentially optimal in a worst-case sense, and prior to this work the best known algorithms even for the case $r = 4$ ran in exponential time.

We will focus on a special case to illustrate the basic idea behind variable reduction. Suppose that $\text{rank}(M) = r$, and our goal is to decide whether or not $\text{rank}^+(M) = r$. This is called the simplicial factorization problem. Can we find an alternate system of polynomial inequalities that expresses this decision problem but uses many fewer variables? The following simple but useful observation will pave the way:

Claim 2.2.6 *In any solution to the simplicial factorization problem, A and W must have full column and row rank, respectively.*

Proof: If $M = AW$, then the column span of A must contain the columns of M, and similarly, the row span of W must contain the rows of M. Since $\text{rank}(M) = r$, we conclude that A and W must have r linearly independent columns and rows, respectively. Since A has r columns and W has r rows, this implies the claim. ■

Hence we know that A has a left pseudo-inverse A^+ and W has a right pseudo-inverse W^+ so that $A^+A = WW^+I_r$, where I_r is the $r \times r$ identity matrix. We will make use of these pseudo-inverses to reduce the number of variables in our system of polynomial inequalities. In particular:

$$A^+AW = W$$

so we can recover the columns of W from a linear transformation of the columns of M. Similarly, we can recover the rows of A from a linear transformation of the rows of M. This leads to the following alternative system of polynomial inequalities:

$$\begin{cases} MW^+A^+M = M \\ MW^+ \geq 0 \\ A^+M \geq 0 \end{cases} \tag{2.2}$$

A priori, it is not clear that we have made progress, since this system also has $nr + mr$ variables corresponding to the entries of A^+ and W^+. However, consider the matrix MW^+. If we represent S^+ as an $n \times r$ matrix, then we are describing its action on all vectors, but the crucial observation is that we only

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

need to know how S^+ acts on the rows of M that span an r-dimensional space. Hence we can apply a change of basis to write

$$M_C = MU$$

where U is an $n \times r$ matrix that has a right pseudo-inverse. Similarly, we can write

$$M_R = VM$$

where V is an $r \times m$ matrix that has a left pseudo-inverse. Now we get a new system:

$$\begin{cases} M_C S T M_R = M \\ M_C S \geq 0 \\ T M_R \geq 0 \end{cases} \tag{2.3}$$

Notice that S and T are both $r \times r$ matrices, and hence there are $2r^2$ variables in total. Moreover, this formulation is equivalent to the simplicial factorization problem in the following sense:

Claim 2.2.7 *If $rank(M) = rank^+(M) = r$, then (2.3) has a solution.*

Proof: Using the notation above, we can set $S = U^+W^+$ and $T = A^+V^+$. Then $M_C S = MUU^+W^+ = A$ and similarly $TM_R = A^+V^+VM = W$, and this implies the claim. ■

This is often called *completeness*, since if there is a solution to the original problem, we want there to be a valid solution to our reformulation. We also need to prove *soundness*, that any solution to the reformulation yields a valid solution to the original problem:

Claim 2.2.8 *If there is a solution to (2.3), then there is a solution to (2.1).*

Proof: For any solution to (2.3), we can set $A = M_C S$ and $W = TM_R$, and it follows that $A, W \geq 0$ and $M = AW$. ■

It turns out to be quite involved to extend the ideas above to nonnegative matrix factorization in general. The main idea in [112] is to first establish a new normal form for nonnegative matrix factorization, and use the observation that even though A could have exponentially many maximal sets of linearly independent columns, their psueudo-inverses are algebraically dependent and can be expressed over a common set of r^2 variables using Cramer's rule. Additionally, Arora et al. [13] showed that any algorithm that solves even the simplicial factorization problem in $(nm)^{o(r)}$ time yields a subexponential time

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

algorithm for 3-SAT, and hence the algorithms above are nearly optimal under standard complexity assumptions.

Further Remarks

Earlier in this section, we gave a simple example that illustrates a separation between the rank and the nonnegative rank. In fact, there are more interesting examples of separations that come up in theoretical computer science, where a natural question is to express a particular polytope P in n dimensions, which has exponentially many facets as the projection of a higher dimensional polytope Q with only polynomially many facets. This is called an *extended formulation*, and a deep result of Yannakakis is that the minimum number of facets of any such Q – called the *extension complexity* of P – is precisely equal to the nonnegative rank of some matrix that has to do with the geometric arrangement between vertices and facets of P [144]. Then the fact that there are explicit polytopes P whose extension complexity is exponential is intimately related to finding explicit matrices that exhibit large separations between their rank and nonnegative rank.

Furthermore, the nonnegative rank also has important applications in communication complexity, where one of the most important open questions – the *log-rank conjecture* [108] – can be reformulated by asking: Given a Boolean matrix M, is $\log \text{rank}^+(M) \leq (\log \text{rank}(M))^{O(1)}$? Thus, in the example above, the fact that the nonnegative rank cannot be bounded by any function of the rank could be due to the entries of M taking on many distinct values.

2.3 Stability and Separability

Here we will give a geometric (as opposed to algebraic) interpretation of nonnegative matrix factorization that will offer new insights into why it is hard in the worst case and what types of features make it easy. In particular, we will move beyond worst-case analysis and work with a new assumption called *separability* that will allow us to give an algorithm that runs in polynomial time (even for large values of r). This assumption was first introduced to understand conditions under which the nonnegative matrix factorization problem has a unique solution [65], and this is a common theme in algorithm design

Theme 1 *Looking for cases where the solution is unique and robust will often point to cases where we can design algorithms with provable guarantees in spite of worst-case hardness results.*

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Cones and Intermediate Simplicies

Here we will develop some geometric intuition about nonnegative matrix factorization – or, rather, an important special case of it called simplicial factorization that we introduced in the previous section. First, let us introduce the notion of a cone:

Definition 2.3.1 *Let A be an $m \times r$ matrix. Then the cone generated by the columns of A is*

$$\mathcal{C}_A = \{Ax | x \geq 0\}.$$

We can immediately connect this to nonnegative matrix factorization.

Claim 2.3.2 *Given matrix M, A of dimension $m \times n$ and $m \times r$, respectively, there is an entrywise nonnegative matrix W of dimension $r \times n$ with $M = AW$ if and only if $\mathcal{C}_M \subseteq \mathcal{C}_A$.*

Proof: In the forward direction, suppose $M = AW$, where W is entrywise nonnegative. Then any vector $y \in \mathcal{C}_M$ can be written as $y = Mx$ where $x \geq 0$, and then $y = AWx$ and the vector $Wx \geq 0$, and hence $y \in \mathcal{C}_A$ too. In the reverse direction, suppose $\mathcal{C}_M \subseteq \mathcal{C}_A$. Then any column $M_i \in \mathcal{C}_A$ and we can write $M_i = AW_i$ where $W_i \geq 0$. Now we can set W to be the matrix whose columns are $\{W_i\}_i$ and this completes the proof. ■

What makes nonnegative matrix factorization difficult is that both A and W are unknown (if one were known, say A, then we could solve for the other by setting up an appropriate linear program, which amounts to representing each column of M in $\mathcal{C}_A$).

Vavasis [139] was the first to introduce the simplicial factorization problem, and one of his motivations was that it turns out to be connected to a purely geometric problem about fitting a simplex in between two given polytopes. This is called the intermediate simplex problem:

Definition 2.3.3 *An instance of the intermediate simplex problem consists of P and Q with $P \subseteq Q \subseteq \mathbb{R}^{r-1}$, and P is specified by its vertices and Q is specified by its facets. The goal is to find a simplex K with $P \subseteq K \subseteq Q$.*

In the next section we will show that the simplicial factorization problem and the intermediate simplex problem are equivalent.

Reductions

We will prove that the simplicial factorization problem and the intermediate simplex problem are equivalent in the sense that there is a polynomial time

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

reduction in both directions. We will do so by way of a few intermediate problems.

Suppose we are given an instance of the simplicial factorization problem. Then we can write $M = UV$, where U and V have inner dimension r but are not necessarily entrywise nonnegative. If we can find an invertible $r \times r$ matrix T where UT and $T^{-1}V$ are both entrywise nonnegative, then we have found a valid nonnegative matrix factorization with inner dimension r.

Claim 2.3.4 *If rank$(M) = r$ and $M = UV$ and $M = AW$ are two factorizations that have inner-dimension r, then*

(1) colspan(U) = colspan(A) = colspan(M) and
(2) rowspan(V) = rowspan(W) = rowspan(M).

This follows from basic facts in linear algebra, and implies that any two such factorizations $M = UV$ and $M = AW$ can be linearly transformed into each other via some invertible $r \times r$ matrix T. Hence the intermediate simplex problem is equivalent to

Definition 2.3.5 *An instance of the problem **P1** consists of an $m \times n$ entrywise nonnegative matrix M with rank$(M) = r$ and $M = UV$ with inner dimension r. The goal is to find an invertible $r \times r$ matrix where both UT and $T^{-1}V$ are entrywise nonnegative.*

Caution: The fact that you can start out with an arbitrary factorization and ask to rotate it into a nonnegative matrix factorization of minimum inner dimension but haven't painted yourself into a corner is particular to the simplicial factorization problem only! It is generally not true when rank$(M) <$ rank$^+(M)$.

Now we can give a geometric interoperation of **P1**:

(1) Let $u_1, u_2, \ldots, u_m$ be the rows of U.
(2) Let $t_1, t_2, \ldots, t_r$ be the columns of T.
(3) Let $v_1, v_2, \ldots, v_n$ be the columns of V.

We will first work with an intermediate cone problem, but its connection to the intermediate simplex problem will be immediate. Toward that end, let P be the cone generated by $u_1, u_2, \ldots, u_m$, and let K be the cone generated by $t_1, t_2, \ldots, t_r$. Finally, let Q be the cone given by

$$Q = \{x | \langle u_i, x \rangle \geq 0 \text{ for all } i\}.$$

It is not hard to see that Q is a cone in the sense that it is generated as all nonnegative combinations of a finite set of vectors (its extreme rays), but we

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

have instead chosen to represent it by its supporting hyperplanes (through the origin).

Claim 2.3.6 *UT is entrywise nonnegative if and only if $\{t_1, t_2, \ldots, t_r\} \subseteq Q$.*

This follows immediately from the definition of Q, because the rows of U are its supporting hyperplanes (through the origin). Hence we have a geometric reformulation of the constraint UT that is entrywise nonnegative in **P1**. Next, we will interpret the other constraint, that $T^{-1}V$ is entrywise nonnegative too.

Claim 2.3.7 *$T^{-1}V$ is entrywise nonnegative if and only if $\{v_1, v_2, \ldots, v_m\} \subseteq K$.*

Proof: Consider $x_i = T^{-1}v_i$. Then $Tx_i = T(T^{-1})v_i = v_i$, and hence x_i is a representation of v_i as a linear combination of $\{t_1, t_2, \ldots, t_r\}$. Moreover, it is the unique representation, and this completes the proof. ■

Thus **P1** is equivalent to the following problem:

Definition 2.3.8 *An instance of the intermediate cone problem consists of cones P and Q with $P \subseteq Q \subseteq \mathbb{R}^{r-1}$, and P is specified by its extreme rays and Q is specified by its supporting hyperplanes (through the origin). The goal is to find a cone K with r extreme rays and $P \subseteq K \subseteq Q$.*

Furthermore, the intermediate cone problem is easily seen to be equivalent to the intermediate simplex problem by intersecting the cones in it with a hyperplane, in which case a cone with extreme rays becomes a convex hull of the intersection of those rays with the hyperplane.

Geometric Gadgets

Vavasis made use of the equivalences in the previous section to construct certain geometric gadgets to prove that nonnegative matrix factorization is *NP*-hard. The idea was to construct a two-dimensional gadget where there are only two possible intermediate triangles, which can then be used to represent the truth assignment for a variable x_i. The description of the complete reduction and the proof of its soundness are involved (see [139]).

Theorem 2.3.9 *[139] Nonnegative matrix factorization, simplicial factorization, intermediate simplex, intermediate cone, and **P1** are all NP-hard.*

Arora et al. [13] improved upon this reduction by constructing low-dimensional gadgets with many more choices. This allows them to reduce from the d-SUM problem, where we are given a set of n numbers and the goal is to find a set of d of them that sum to zero. The best known algorithms for this

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

problem run in time roughly $n^{\lceil d/2 \rceil}$. Again, the full construction and the proof of soundness are involved.

Theorem 2.3.10 *Nonnegative matrix factorization, simplicial factorization, intermediate simplex, intermediate cone, and* ***P1*** *all require time at least* $(nm)^{\Omega(r)}$ *unless there is a subexponential time algorithm for 3-SAT.*

In all of the topics we will cover, it is important to understand what makes the problem hard in order to identify what makes it easy. The common feature in all of the above gadgets is that the gadgets themselves are highly unstable and have multiple solutions, and so it is natural to look for instances where the answer itself is robust and unique in order to identify instances that can be solved more efficiently than in the worst case.

Separability

In fact, Donoho and Stodden [64] were among the first to explore the question of what sorts of conditions imply that the nonnegative matrix factorization of minimum inner dimension is unique. Their original examples came from toy problems in image segmentation, but it seems like the condition itself can most naturally be interpreted in the setting of text analysis.

Definition 2.3.11 *We call A separable if, for every column i of A, there is a row j where the only nonzero is in the* i^{th} *column. Furthermore, we call j an anchor word for column i.*

In fact, separability is quite natural in the context of text analysis. Recall that we interpret the columns of A as topics. We can think of separability as the promise that these topics come with *anchor words*; informally, for each topic there is an unknown anchor word, and if it occurs in a document, the document is (partially) about the given topic. For example, *401k* could be an anchor word for the topic *personal finance*. It seems that natural language contains many such highly specific words.

We will now give an algorithm for finding the anchor words and for solving instances of nonnegative matrix factorization where the unknown A is separable in polynomial time.

Theorem 2.3.12 *[13] If* $M = AW$ *and* A *is separable and* W *has full row rank, then the* **Anchor Words Algorithm** *outputs* A *and* W *(up to rescaling).*

Why do anchor words help? It is easy to see that if A is separable, then the rows of W appear as rows of M (after scaling). Hence we just need to determine which rows of M correspond to anchor words. We know from our discussion in Section 2.3 that if we scale M, A, and W so that their rows sum to one, the

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

convex hull of the rows of W contains the rows of M. But since these rows appear in M as well, we can try to find W by iteratively deleting rows of M that do not change its convex hull.

Let M^i denote the i^{th} row of M and let M^I denote the restriction of M to the rows in I for $I \subseteq [n]$. So now we can find the anchor words using the following simple procedure:

Find Anchors [13]

Input: matrix $M \in \mathbb{R}^{m \times n}$ satisfying the conditions in Theorem 2.3.12
Output: $W = M^I$

Delete duplicate rows:
Set $I = [n]$
For $i = 1, 2, \ldots, n$
    If $M^i \in \text{conv}(\{M^j | j \in I, j \neq i\})$, set $I \leftarrow I - \{i\}$
End

Here in the first step, we want to remove redundant rows. If two rows are scalar multiples of each other, then one being in the cone generated by the rows of W implies the other is too, so we can safely delete one of the two rows. We do this for all rows, so that in the equivalence class of rows that are scalar multiples of each other, exactly one remains. We will not focus on this technicality in our discussion, though.

It is easy to see that deleting a row of M that is not an anchor word will not change the convex hull of the remaining rows, and so the above algorithm terminates with a set I that only contains anchor words. Moreover, at termination

$$\text{conv}(\{M^i | i \in I\}) = \text{conv}(\{M^j\}_j).$$

Alternatively, the convex hull is the same as at the start. Hence the anchor words that are deleted are redundant and we can just as well do without them.

Anchor Words [13]

Input: matrix $M \in \mathbb{R}^{n \times m}$ satisfying the conditions in Theorem 2.3.12
Output: A, W

Run **Find Anchors** on M, let W be the output
Solve for nonnegative A that minimizes $\|M - AW\|_F$ (convex programming)
End

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

The proof of theorem follows immediately from the proof of correctness of **Find Anchors** and the fact that $\text{conv}(\{M^i\}_i) \subseteq \text{conv}(\{W^i\}_i)$ if and only if there is a nonnegative A (whose rows sum to one) with $M = AW$.

The above algorithm, when naively implemented, would be prohibitively slow. Instead, there have been many improvements to the algorithm ([33], [100], [78]), and we will describe one in particular that appears in [12]. Suppose we choose a row M^i at random. Then it is easy to see that the farthest row from M^i will be an anchor word.

Similarly, if we have found one anchor word, the farthest row from it will be another anchor word, and so on. In this way we can greedily find all of the anchor rows, and moreover, this method only relies on pairwise distances and projection, so we can apply dimension reduction before running this greedy algorithm. This avoids linear programming altogether in the first step in the above algorithm, and the second step can also be implemented quickly, because it involves projecting a point into a $k-1$-dimensional simplex.

2.4 Topic Models

In this section, we will work with stochastic models to generate a collection of documents. These models are called *topic models*, and our goal is to learn their parameters. There is a wide range of types of topic models, but all of them fit into the following abstract framework:

Abstract Topic Model

Parameters: topic matrix $A \in \mathbb{R}^{m \times r}$, distribution μ on the simplex in $\mathbb{R}^r$

For $i = 1$ to n
    Sample W_i from μ
    Generate L words by sampling i.i.d. from the distribution AW_i
End

This procedure generates n documents of length L, and our goal is to infer A (and μ) from observing samples from this model. Let $\widetilde{M}$ be the observed term-by-document matrix. We will use this notation to distinguish it from its expectation

$$\mathbb{E}[\widetilde{M}|W] = M = AW.$$

In the case of nonnegative matrix factorization, we were given M and not $\widetilde{M}$. However, these matrices can be quite far apart! Hence, even though

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

each document is described as a distribution on words, we only have partial knowledge of this distribution in the form of L samples from it. Our goal is to design algorithms that provably work even in these challenging models.

Now is a good time to point out that this model contains many well-studied topic models as a special case. All of them correspond to different choices of μ, the distribution that is used to generate the columns of W. Some of the most popular variants are:

(a) **Pure Topic Model:** Each document is about only one topic, hence μ is a distribution on the vertices of the simplex and each column in W has exactly one nonzero.
(b) **Latent Dirichlet Allocation [36]:** μ is a Dirichlet distribution. In particular, one can generate a sample from a Dirichlet distribution by taking independent samples from r (not necessarily identical) gamma distributions and then renormalizing so that their sum is one. This topic model allows documents to be about more than one topic, but its parameters are generally set so that it favors relatively sparse vectors W_i.
(c) **Correlated Topic Model [35]:** Certain pairs of topics are allowed to be positively or negatively correlated, and μ is constrained to be log-normal.
(d) **Pachinko Allocation Model [105]:** This is a multilevel generalization of LDA that allows for certain types of structured correlations.

In this section, we will use our algorithm for separable nonnegative matrix factorization to provably learn the parameters of a topic model for (essentially) any topic model where the topic matrix is separable. Thus this algorithm will work even in the presence of complex relationships between the topics.

The Gram Matrix

In this subsection, we will introduce two matrices, G and R, which we will call the Gram matrix and the topic co-occurrence matrix, respectively. The entries of these matrices will be defined in terms of the probability of various events. And throughout this section, we will always have the following experiment in mind: We generate a document from the abstract topic model, and let w_1 and w_2 denote the random variables for its first and second word, respectively. With this experiment in mind, we can define the Gram matrix:

Definition 2.4.1 *Let G denote the $m \times m$ matrix where*

$$G_{j,j'} = \mathbb{P}[w_1 = j, w_2 = j'].$$

Moreover, for each word, instead of sampling from AW_i, we can sample from W_i to choose which column of A to sample from. This procedure still

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

generates a random sample from the same distribution AW_i, but each word $w_1 = j$ is annotated with the topic from which it came, $t_1 = i$ (i.e., the column of A we sampled it from). We can now define the topic co-occurrence matrix:

Definition 2.4.2 *Let R denote the $r \times r$ matrix where*

$$R_{i,i'} = \mathbb{P}[t_1 = i, t_2 = i'].$$

Note that we can estimate the entries of G directly from our samples, but we cannot directly estimate the entries of R. Nevertheless, these matrices are related according to the following identity:

Lemma 2.4.3 $G = ARA^T$

Proof: We have

$$\begin{aligned} G_{j,j'} &= \mathbb{P}[w_1 = j, w_2 = j'] = \sum_{i,i'} \mathbb{P}[w_1 = j, w_2 = j' | t_1 = i, t_2 = i'] \mathbb{P}[t_1 = i, t_2 = i'] \\ &= \sum_{i,i'} \mathbb{P}[w_1 = j | t_1 = i] \mathbb{P}[w_2 = j' | t_2 = i'] \mathbb{P}[t_1 = i, t_2 = i'] \\ &= \sum_{i,i'} A_{j,i} A_{j',i'} R_{i,i'} \end{aligned}$$

where the second-to-last line follows, because conditioned on their topics, w_1 and w_2 are sampled independently from the corresponding columns of A. This completes the proof. ■

The crucial observation is that $G = A(RA^T)$ where A is separable and RA^T is nonnegative. Hence if we renormalize the rows of G to sum to one, the anchor words will be the extreme points of the convex hull of all of the rows and we can identify them through our algorithm for separable nonnegative matrix factorization. Can we infer the rest of A?

Recovery via Bayes's Rule

Consider the posterior distribution $\mathbb{P}[t_1 | w_1 = j]$. This is the posterior distribution on which topic generated $w_1 = j$ when you know nothing else about the document. The posterior distributions are just renormalizations of A so that the rows sum to one. Then suppose j is an anchor word for topic i. We will use the notation $j = \pi(i)$. It is easy to see

$$\mathbb{P}[t_1 = i' | w_1 = \pi(i)] = \begin{cases} 1, \text{ if } i' = i \\ 0 \text{ else.} \end{cases}$$

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Now we can expand:

$$\mathbb{P}[w_1 = j | w_2 = j'] = \sum_{i'} \mathbb{P}[w_1 = j | w_2 = j', t_2 = i'] \mathbb{P}[t_2 = i' | w_2 = j']$$
$$= \sum_{i'} \mathbb{P}[w_1 = j | t_2 = i'] \mathbb{P}[t_2 = i' | w_2 = j']$$

In the last line we used the following identity:

Claim 2.4.4 $\mathbb{P}[w_1 = j | w_2 = j', t_2 = i'] = \mathbb{P}[w_1 = j | t_2 = i']$

We leave the proof of this claim as an exercise. We will also use the identity below:

Claim 2.4.5 $\mathbb{P}[w_1 = j | t_2 = i'] = \mathbb{P}[w_1 = j | w_2 = \pi(i')]$

Proof:

$$\mathbb{P}[w_1 = j | w_2 = \pi(i')] = \sum_{i''} \mathbb{P}[w_1 = j | w_2 = \pi(i'), t_2 = i''] \mathbb{P}[t_2 = i'' | w_2 = \pi(i')]$$
$$= \mathbb{P}[w_1 = j | w_2 = \pi(i'), t_2 = i']$$

where the last line follows, because the posterior distribution on the topic $t_2 = i''$, given that w_2 is an anchor word for topic i', is equal to one if and only if $i'' = i'$. Finally, the proof follows by invoking Claim 2.4.4. ■

Now we can proceed:

$$\mathbb{P}[w_1 = j | w_2 = j'] = \sum_{i'} \mathbb{P}[w_1 = j | w_2 = \pi(i')] \underbrace{\mathbb{P}[t_2 = i' | w_2 = j']}_{\text{unknowns}}$$

Hence this is a linear system in variables $\mathbb{P}[w_1 = j | w_2 = \pi(i')]$ and it is not hard to show that if R has full rank, then it has a unique solution.

Finally, by Bayes's rule we can compute the entries of A:

$$\mathbb{P}[w = j | t = i] = \frac{\mathbb{P}[t = i | w = j] \mathbb{P}[w = j]}{\mathbb{P}[t = i]}$$
$$= \frac{\mathbb{P}[t = i | w = j] \mathbb{P}[w = j]}{\sum_{j'} \mathbb{P}[t = i | w = j'] \mathbb{P}[w = j']}$$

And putting it all together, we have the following algorithm:

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Recover [14], [12]

Input: term-by-document matrix $M \in \mathbb{R}^{n \times m}$
Output: A, R

Compute the Gram matrix G
Compute the anchor words via **Separable NMF**
Solve for $\mathbb{P}[t = i|w = j]$
Compute $\mathbb{P}[w = j|t = i]$ from Bayes's rule

Theorem 2.4.6 *[14] There is a polynomial time algorithm to learn the topic matrix for any separable topic model, provided that R is full rank.*

Remark 2.4.7 *The running time and sample complexity of this algorithm depend polynomially on* $m, n, r, \sigma_{min}(R), p, 1/\epsilon, \log 1/\delta$ *where* p *is a lower bound on the probability of each anchor word,* ϵ *is the target accuracy, and* δ *is the failure probability.*

Note that this algorithm works for short documents, even for $L = 2$.

Experimental Results

Now we have provable algorithms for nonnegative matrix factorization and topic modeling under separability. But are *natural* topic models separable or close to being separable? Consider the following experiment:

(1) **UCI Dataset**: A collection of 300,000 *New York Times* articles
(2) **MALLET**: A popular topic-modeling toolkit

We trained MALLET on the UCI dataset and found that with $r = 200$, about 0.9 fraction of the topics had a near anchor word – i.e., a word where $\mathbb{P}[t = i|w = j]$ had a value of at least 0.9 on some topic. Indeed, the algorithms we gave can be shown to work in the presence of some modest amount of error – deviation from the assumption of separability. But can they work with this much modeling error?

We then ran the following additional experiment:

(1) Run MALLET on the UCI dataset, learn a topic matrix ($r = 200$).
(2) Use A to generate a new set of documents synthetically from an LDA model.

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

(3) Run MALLET and our algorithm on a new set of documents, and compare their outputs to the ground truth. In particular, compute the minimum cost matching between the columns of the estimate and the columns of the ground truth.

It is important to remark that this is a biased experiment – biased *against* our algorithm! We are comparing how well we can find the hidden topics (in a setting where the topic matrix is only close to separable) to how well MALLET can find its own output again. And with enough documents, we can find it more accurately and hundreds of times faster! This new algorithm enables us to explore much larger collections of documents than ever before.

2.5 Exercises

Problem 2-1: Which of the following are equivalent definitions of nonnegative rank? For each, give a proof or a counterexample.

(a) The smallest r such that M can be written as the sum of r rank-one nonnegative matrices
(b) The smallest r such that there are r nonnegative vectors $v_1, v_2, \ldots, v_r$ such that the cone generated by them contains all the columns of M
(c) The largest r such that there are r columns of M, $M_1, M_2, \ldots, M_r$ such that no column in the set is contained in the cone generated by the remaining $r - 1$ columns

Problem 2-2: Let $M \in \mathbb{R}^{n \times n}$ where $M_{i,j} = (i - j)^2$. Prove that $\text{rank}(M) = 3$ and that $\text{rank}^+(M) \geq \log_2 n$. *Hint:* To prove a lower bound on $\text{rank}^+(M)$, it suffices to consider just where it is zero and where it is nonzero.

Problem 2-3: Papadimitriou et al. [118] considered the following document model: $M = AW$ and each column of W has only one nonzero and the support of each column of A is disjoint. Prove that the left singular vectors of M are the columns of A (after rescaling). You may assume that all the nonzero singular values of M are distinct. *Hint:* MM^T is a block diagonal after applying a permutation π to its rows and columns.

Problem 2-4: Consider the following algorithm:

Greedy Anchor Words [13]

Input: matrix $M \in \mathbb{R}^{n \times m}$ satisfying the conditions in Theorem 2.3.12
Output: A, W

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press

Set $S = \emptyset$
For $i = 2$ to r
    Project the rows of M orthogonal to the span of vectors in S
    Add the row with the largest ℓ_2 norm to S
End

Let $M = AW$ where A is separable and the *rows* of M, A, and W are normalized to sum to one. Also assume W has full row rank. Prove that Greedy Anchor Words finds all the anchor words and nothing else. *Hint:* the ℓ_2 norm is strictly convex — i.e., for any $x \neq y$ and $t \in (0, 1)$, $\|tx + (1-t)y\|_2 < t\|x\|_2 + (1-t)\|y\|_2$.

https://doi.org/10.1017/9781316882177.003 Published online by Cambridge University Press