Universität des Saarlandes
Max-Planck-Institut für Informatik
AG5

On Some Problems of Rounding Rank

Master's Thesis in Computer Science
by

Stefan Neumann

supervised by

Dr. Pauli Miettinen

Prof. Dr. Rainer Gemulla

reviewers

Dr. Pauli Miettinen

Prof. Dr. Gerhard Weikum

September 2015

Hilfsmittelerklärung

Hiermit versichere ich, die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben.

Non-plagiarism Statement

Hereby I confirm that this thesis is my own work and that I have documented all sources used.

Saarbrücken, den 29. September 2015,

(Stefan Neumann)

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlichtwird.

Declaration of Consent

Herewith I agree that my thesis will be made available through the library of the Computer Science Department at Saarland University.

Saarbrücken, den 29. September 2015,

(Stefan Neumann)

Katten
sit i tunet
når du kjem.
Snakk litt med katten.
Det er han som er varast i garden.

- Olav H. Hauge

Je n'aime pas ce monde. Décidément, je ne l'aime pas. La société dans laquelle je vis me dégoûte; le publicite m'éœure; l'informatique me fait vomir. [...] Ce monde a besoin de tout, sauf d'informations supplémentaires.

- Michel Houellebecq, Exentension du domaine de la lutte

Abstract

This thesis is devoted to the study of the *rounding rank problem*: Given a binary matrix and a real number, the *rounding threshold*, we want to find the real-valued matrix of lowest rank, that after rounding according to the given threshold results in the given input matrix. We call this rank the *rounding rank*.

Using the theory of hyperplane arrangements, we prove that rounding rank is polynomial-time equivalent to finding the smallest-dimensional Euclidean space, such that we can separate certain subsets of points by affine hyperplanes.

We also tightly characterise the role of the rounding threshold. The results show that changing the rounding threshold can increase the rounding rank only by a constant. We further classify when this happens.

The thesis also contains two algorithms that heuristically compute approximations of the rounding rank. The first algorithm is motivated by the Eckart–Young Theorem and is based on truncated singular value decomposition. The second algorithm is a randomised algorithm, which uses intuition from hyperplane arrangements and applies linear programming. Both algorithms were tested on synthetic and on real-world data.

Rounding rank is closely related to sign rank. In this thesis we give the first comprehensive summary of the existing literature on this topic. In addition, for the first time sign rank and work on the geometric representation of graphs are related to one another.

Acknowledgements

First and foremost I would like to express my deep gratitude towards my supervisor Pauli Miettinen for providing me with this interesting and challenging thesis topic. Your constant support and your feedback were invaluable to me and without them this work would be much worse. I will always have fond memories of the interesting conversations we had.

Rainer Gemulla's detailed comments on the proofs in this thesis were much appreciated and helped me a lot to improve this text.

Further thanks go to Gerhard Weikum for being a part of the thesis committee.

Jilles, over the last year you were a superb mentor and I do not want to miss a single of our discussions about research, science or life in general. They will not be forgotten.

Kailash, you were the nicest and most courteous person I could have shared an office with. Thank you for these good times. Your feedback on this thesis made it much more accessible and improved its quality a lot. Thank you!

EDA group, you are very cool! For providing me with an office. For hanging out with me at the EDA events. For ordering Mate with me. And for bearing my occasional grumpiness.

Thank you, GradSchool, for my stipend and for giving me the opportunity to gain first research experiences.

Contents

Chapter 1

Introduction

In the real-world we often encounter binary data, for example in transaction databases or in presence/absence data. Hence, in data mining we are highly interested in finding techniques for the extraction of patterns from this data [Han et al., 2011]. Unfortunately, due to the discreteness of the data, most algorithms that operate on these inputs have to solve combinatorial optimisation problems, which in many cases are not only NP-hard to solve exactly, but also hard to approximate [Trevisan, 2004].

To avoid these issues of computational hardness, one option is to consider the data sets as matrices and then to use standard techniques from real-valued numerical analysis. Some of them, like singular value decomposition, proved to be useful in data science (see, e.g. Leskovec et al. [2014]). On the upside, the real-valued methods are well-known, posses good theoretical properties and they are provided out of the box by many software libraries. Also, they can be computed in polynomial time and in practise they can be implemented highly efficiently [Golub and Van Loan, 1996]. The downside is that they only provide floating-point results, which are sometimes hard to relate to the binary input data. Furthermore, the real-valued matrix factorisations of sparse input matrices are often dense. This makes them hard to interpret in pratise.

A compromise to avoid the previously mentioned issues is to first use real-valued methods and then to use *rounding* to ensure the desired properties of the results. For example, Achlioptas and McSherry [2001] and Drineas et al. [2015] use rounding to ensure the sparsity of the outputs of their algorithms. Erdős et al. [2014] use singular value decomposition and rounding according to a deterministic threshold in order to get results from the binary domain. In practise, it turns out that these approaches deliver very good results.

It is interesting to raise the question, whether we can theoretically justify that such approaches perform so well in practice. This thesis is devoted to the question: *How powerful is the rounding?*

In order to answer this question, we formally define a novel matrix rank on binary matrices that measures the gain that we can obtain from using rounding. In more detail, this thesis introduces the *rounding rank*: Given a binary matrix, its *rounding rank* is the smallest rank of any real-valued matrix that after rounding according to some deterministic threshold results in the input matrix.

To relate this theoretical concept to the posed question 'How powerful is the rounding?', let us make the following observation: If the rounding is very powerful, then the rounding rank of any binary matrix should be significantly smaller than its standard rank or its Boolean rank. On the other hand, if the rounding rank and the other ranks are very close in many cases, then the rounding does not provide us with much gain.

This thesis seeks to mathematically explore the rounding rank and its properties.

We will discuss two equivalent problems, which are polynomial-time equivalent to the rounding rank. The first of these problems is finding the smallest-dimensional Euclidean space, such that we can separate certain subsets of points by affine hyperplanes. The second problem we will discuss is a problem of combinatorial geometry, which uses hyperplane arrangements.

We will further prove that the threshold according to which we round has only a minor effect on the rounding rank of a matrix.

It turns out that the rounding rank is closely related to the *sign rank*, which was researched in the communities of communication complexity and learning theory since the 1980s. This thesis provides a comprehensive summary of the existing literature on sign rank.

Near the end of this thesis we will give two algorithms that compute approximations of the rounding rank of a binary matrix. One of them is a greedy algorithm, that makes use of the truncated singular value decomposition and the Eckart–Young Theorem. The other one is a randomised approximation algorithm, that first solves a problem of combinatorial geometry and then computes a rounding rank decomposition from it.

Both algorithms were tested on synthetic and on real-world data. While we were able to show that in theory rounding rank is NP-hard to compute and that almost all matrices have a large rounding rank, the experiments show that for many matrices their rounding rank is significantly less than their real rank. Particularly for real-world data we were able to show that its rounding ranks were much smaller than its real ranks.

Contributions

Let us shortly summarise which contributions were made in this thesis.

The author derived the characterisations of rounding rank and sign rank as given in Theorem 4.3 and in Theorem 5.2 independently of Paturi and Simon [1984] before learning about their results.

The work on sign rank which is presented in this thesis was done in different communities, such as communication complexity or learning theory. Those different fields used different notations to state their results and they also had different motivations for their research. This thesis is the first work that coherently and comprehensively presents all of these results in a single volume. In addition, this is the first time the results of Kang and Müller [2012] are considered in the context of sign rank.

To the author's best knowledge this is the first time that the notion of *thresholdable* matrices as in Definition 5.6 is discussed and used to provide the novel results from Theorem 5.8 and Corollary 5.9, which characterise the role of the rounding threshold.

The results from Theorems 4.3 and 5.13 were known in the sign rank community, but in this thesis it is the first time that they are stated in this form and that their proofs are presented with full details.

The algorithms that are discussed in Chapter 7 are new. It is the first time algorithms for the rounding rank problem are presented and practically implemented. They can also be used to solve the sign rank problem. For the latter, to the author's best knowledge they present the first implementations of algorithms approximating the sign rank.

Basic Notation

This section shortly explains the basic notation and naming conventions that are used in this thesis.

We will often encounter matrices, vectors and scalars in the following chapters. To make them easier to separate, matrices will be denoted by bold-faced capital letters like $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ or $\boldsymbol{B} \in \{0,1\}^{m \times n}$. Vectors are usually column vectors and will be referred to using bold small letters, for example $\boldsymbol{x} \in \mathbb{R}^d$ or $\boldsymbol{y} \in \mathbb{R}^d$. For scalars we will use *standard* small letters, e.g. $c \in \mathbb{R}$.

In most cases we will be arguing about the rows of matrices; then for a matrix $\boldsymbol{A}$ with n rows its i'th row will be given by $\boldsymbol{a}_i$.

The *dot product* is the function $\langle \cdot, \cdot \rangle : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, which for two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ with components $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d)$ and $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_d)$ is defined by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{d} \boldsymbol{x}_i \boldsymbol{y}_i$.

Outline of the Thesis

Let us conclude the introduction by providing a short outline of how this thesis is arranged.

The next two chapters provide the necessary theoretical background for the rest of the thesis. In order to do this they define two well-knowns ranks. Chapter 2 introduces the standard rank for real-valued matrices as it is taught in linear algebra courses. Chapter 3 discusses the somewhat less well-known Boolean rank for binary matrices, which gained increasing recognition in the field of data mining over the last decade.

The following Chapter 4 introduces the *sign rank* of a binary matrix, which can be seen as a combination of both standard rank and Boolean rank. This rank was studied in the communities of communication complexity and learning theory. The chapter gives an extensive summary of the existing literature on this topic and also relates it to the geometrical representation of graphs. Since rounding rank and sign rank are very closely related, this chapter can also be seen as the related work part of this thesis.

Chapter 5 formally defines the *rounding rank* of a binary matrix. Although we will see that this rank is a little bit more general than the sign rank, we will prove that it is not significantly smaller. We will also introduce the *non-negative rounding rank* of a matrix and prove some results for it. The chapter is concluded by a collection of matrices for which we know their exact rounding ranks.

A summary of all four previously introduced different ranks is given in Chapter 6, which compares them with one another in terms of upper and lower bounds and also in terms of computational complexity.

Chapter 7 gives two heuristic algorithms to compute approximations of the rounding ranks of binary matrices. Both algorithms make heavy use of the previously introduced theory. The first algorithm uses the singular value decomposition and insights from the Eckart–Young Theorem. The second algorithm solves a geometrical problem that is equivalent to the rounding rank and uses this result to obtain an approximation. Chapter 8 gives an experimental evaluation of both algorithms on synthetic and on real-world data sets.

The final Chapter 9 summarises the results of this thesis and provides an outlook on further interesting research questions.

Chapter 2

Standard Rank

In this chapter we will recapitulate some basic notions from linear algebra. We will start by discussing linear independence and the rank of real-valued matrices. Also, we will see different characterisations of this rank. After that we will look into the singular value decompositions of matrices, which has many interesting theoretical properties and also turned out to be very useful in practice. We will conclude the chapter by looking into the computational complexity of the methods we introduced.

2.1 Definition and Characterisation

In this section we will formally define the rank of a real-valued matrix. In order to be able to do this, we will need the notion of linear independence.

Definition 2.1 (Linear independence, [see Golub and Van Loan, 1996, page 49])**.** *Let* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in \mathbb{R}^n \setminus \{0\}$*. We call those vectors* linearly dependent*, if there exist scalars* $\alpha_1, \ldots, \alpha_k \in \mathbb{R}$*, such that*

$$\sum_{i=1}^{k} \alpha_i \boldsymbol{v}_i = 0$$

and at least one α_i *is non-zero. If the* $\boldsymbol{v}_i$ *are not linearly dependent, we call them* linearly independent*.*

Having seen this definition, we can now move on to define the rank of a real-valued matrix. Since we will see multiple different ranks in this thesis, we will also refer to it as the *standard rank*.

Definition 2.2 (Standard rank, [see Golub and Van Loan, 1996, page 49]). *Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a real-valued matrix of size $m \times n$. Then the* rank of $\boldsymbol{A}$*, denoted by* $\operatorname{rank}(\boldsymbol{A})$*, is given by the number of linearly independent rows of $\boldsymbol{A}$.*

It turns out that there is more than one way to look at the standard rank of a matrix. The following theorem will give us three different but equivalent ways to look at the rank of a matrix.

Theorem 2.3 (Characterisation of Standard Rank, [see Golub and Van Loan, 1996, page 49]). *Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. Then the following statements are equivalent:*

1. $\operatorname{rank}(\boldsymbol{A}) = k$*, i.e. $\boldsymbol{A}$ has k linearly independent rows.*
2. *$\boldsymbol{A}$ has k linearly independent columns.*
3. *k is the smallest natural number, such that there exist matrices $\boldsymbol{L} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{R} \in \mathbb{R}^{n \times k}$ with $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{R}^T$.*

Sometimes the first point of the theorem is referred to as the *row rank* of the matrix. Similarly, point two is denoted as the *column rank*. Condition number three is also known as the *Schein rank*. According to the theorem all three notions are the same.

Using the first two points of Theorem 2.3 we can derive a trivial upper bound on the rank of a matrix: Given a real-valued $m \times n$ matrix $\boldsymbol{A}$, we have $\operatorname{rank}(\boldsymbol{A}) \leq \min\{m, n\}$. We will say that a matrix with $\operatorname{rank}(\boldsymbol{A}) = \min\{m, n\}$ has *full rank*.

2.2 Singular Value Decomposition

The singular value decomposition (SVD) is one of the most popular matrix decomposition techniques. We will see that it exists for all real-valued matrices and that it has good properties for low-rank approximations of a matrix. Furthermore, it can be computed efficiently.

To introduce the singular value decomposition technically, we need some more terminology. We will call a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ *orthogonal*, if $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}$. Now we can state the following theorem, which proves the existence of the SVD and shows some of its properties.

Theorem 2.4 (Singular Value Decomposition (SVD), [see Golub and Van Loan, 1996, page 70]). *Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a real-valued matrix. Then there exist orthogonal matrices $\boldsymbol{U} \in \mathbb{R}^{m \times m}$ and $\boldsymbol{V} \in \mathbb{R}^{n \times n}$, such that*

$$\boldsymbol{U}^T\boldsymbol{A}\boldsymbol{V} = \operatorname{diag}(\sigma_1, \ldots, \sigma_p) := \boldsymbol{\Sigma} \in \mathbb{R}^{m \times n},$$

where $p = \min\{m, n\}$ *and* $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$.

We will call the scalars σ_i from Theorem 2.4 the *singular values* of $\boldsymbol{A}$. We will write $\sigma_i(\boldsymbol{A})$ to denote the i'th largest singular value of matrix $\boldsymbol{A}$. One can prove that the rank of a matrix is exactly the number of its non-zero singular values (for the proof see Golub and Van Loan [1996, page 71]). The columns from $\boldsymbol{U}$ and $\boldsymbol{V}$ will be called the *left and right singular vectors*, respectively.

Now we will move on to see that the SVD delivers the best low-rank approximation of a matrix and that this approximation is given by the so called truncated SVD. This will be the result of the Eckart–Young Theorem. Before we can state it, we need a bit more notation.

To clarify the notion of "best approximation", we need a notion of the *norm* of a matrix. We will gain it from the *Frobenius norm* $||\cdot||_F$. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, then we define the Frobenius norm of $\boldsymbol{A}$ as

$$||\boldsymbol{A}||_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} \boldsymbol{A}_{ij}^2}.$$

Looking at this formula we see that a matrix has a small Frobenius norm, if its entries are close to zero. In the following we will measure the *distance* of two $m \times n$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ by the Frobenius norm of their difference, i.e. the distance of $\boldsymbol{A}$ and $\boldsymbol{B}$ is given by $||\boldsymbol{A} - \boldsymbol{B}||_F$.

Before we can state the Eckart–Young Theorem, we need to introduce the *truncated SVD*. Let $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ be a real-valued matrix and let $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ be its SVD. Then for $k \leq \min\{m, n\}$ we will denote the matrix with the first k rows of $\boldsymbol{U}$ by $\boldsymbol{U}_{\leq k}$ and the matrix with the first k rows of $\boldsymbol{V}$ by $\boldsymbol{V}_{\leq k}$. Furthermore, we define $\boldsymbol{\Sigma}_{\leq k} = \operatorname{diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k\times k}$. In the following we will call $\boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}\boldsymbol{V}_{\leq k}^T$ the *truncated SVD* of $\boldsymbol{A}$. Notice that the matrix $\boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}\boldsymbol{V}_{\leq k}^T$ has rank exactly k.

Now we can state the fundamental result of Eckart and Young, which proves that the truncated SVD yields the best low-rank approximations of a matrix in terms of the Frobenius norm.

Theorem 2.5 (Eckart–Young, [see Golub and Van Loan, 1996, page 72])**.** *Let* $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ *and let* $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ *be its SVD. Then for all* $k \leq \min\{m, n\}$*,*

$$\min_{\boldsymbol{B}:\operatorname{rank}(\boldsymbol{B})=k} ||\boldsymbol{A} - \boldsymbol{B}||_F = ||\boldsymbol{A} - \boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}\boldsymbol{V}_{\leq k}^T||_F.$$

It is left to discuss the computational complexity of the computation of the singular value decomposition. The SVD of a real-valued $m \times n$ matrix can be computed in time $O(\min\{nm^2, n^2m\})$ (see, e.g. Golub and Van Loan [1996, page 254]). Since the number of non-zero singular values is equal to the rank of a matrix, as we had observed before, we can thus compute the standard rank of matrix in $O(\min\{nm^2, n^2m\})$ by computing the SVD and then counting the number of non-zero singular values.

While in theory it is easy to compute the rank and the SVD of a given matrix, in practise one may have problems with the numerical stability when performing these computations. This is due to problems with the precision of floating point operations in current computers. More on this matter can be found in Paige [1981].

Chapter 3

Boolean Rank

In Chapter 2 we had seen a rank for real-valued matrices. In this chapter we will see another rank, the *Boolean rank*, which is only defined for binary matrices and makes use of their specific properties. We will start by giving a formal definition together with a characterisation as a graph problem. After that we will discuss the computational complexity of computing the Boolean rank and see that it is NP-hard to compute and even hard to approximate.

3.1 Definition and Characterisation

In this section we will introduce the Boolean rank of binary matrices. We will also see that these matrices are equivalent to bipartite graphs and give a characterisation that shows that computing the Boolean rank of a matrix and a certain graph covering problem are equivalent.

In order to define the Boolean rank, we first need to introduce Boolean logic. To this extent let $a, b \in \{0, 1\}$ be binary variables. Then we define the *logical and* of a and b by setting

$$a \wedge b = \begin{cases} 1, & \text{if } a = b = 1, \\ 0, & \text{otherwise.} \end{cases}$$

The *logical or* of a and b is given by

$$a \vee b = \begin{cases} 1, & \text{if } a = 1 \text{ or } b = 1, \\ 0, & \text{otherwise.} \end{cases}$$

For n binary variables $a_1, \ldots, a_n \in \{0,1\}$ we can define the *logical or operator* inductively by setting

$$\bigvee_{k=1}^{1} a_k = a_1$$

and then for $2 \leq m \leq n$ recursively defining

$$\bigvee_{k=1}^{m} a_i = \left(\bigvee_{k=1}^{m-1} a_i \right) \vee a_m.$$

Now we can define Boolean matrix multiplication. Let $\boldsymbol{L} \in \{0,1\}^{m \times k}$ and $\boldsymbol{R} \in \{0,1\}^{n \times k}$ be binary matrices. Then the *Boolean matrix multiplication product*, $\boldsymbol{L} \circ \boldsymbol{R}^T \in \{0,1\}^{m \times n}$, is defined by setting

$$\left(\boldsymbol{L} \circ \boldsymbol{R}^T\right)_{ij} = \bigvee_{l=1}^{k} (\boldsymbol{L}_{il} \wedge \boldsymbol{R}_{jl}),$$

for all entries $i = 1, \ldots, m$ and $j = 1, \ldots, n$ of $\boldsymbol{L} \circ \boldsymbol{R}^T$.

Now we can define the Boolean rank.

Definition 3.1. *Let $\boldsymbol{B} \in \{0,1\}^{m \times n}$ be a binary matrix. The* Boolean rank of $\boldsymbol{B}$ *is the smallest $k \in \mathbb{N}$, such that there exist binary matrices $\boldsymbol{L} \in \{0,1\}^{m \times k}$ and $\boldsymbol{R} \in \{0,1\}^{n \times k}$ with $\boldsymbol{B} = \boldsymbol{L} \circ \boldsymbol{R}^T$. We will denote it by* $\operatorname{brank}(\boldsymbol{B})$.

Having seen this definition we will introduce a characterisation of the Boolean rank, that was given by Monson et al. [1995]. This characterisation will use the fact that there is a one-to-one correspondence between binary matrices and bipartite graphs.

Given a binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$ we can construct its *corresponding bipartite graph* $G_{\boldsymbol{B}} = (A \cup B, E)$ with $A = \{a_1, \ldots, a_m\}$, $B = \{b_1, \ldots, b_n\}$ and $A \cap B = \emptyset$ by setting $(a_i, b_j) \in E$ if and only if $\boldsymbol{B}_{ij} = 1$ for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Notice that in the same way we can construct a binary matrix $\boldsymbol{B}$ from a given bipartite graph. We will also call $\boldsymbol{B}$ the *biadjacency matrix* of $G_{\boldsymbol{B}}$.

To state the characterisation of the Boolean rank, we need two further definitions. Let $G = (A \cup B, E)$ be a bipartite graph. Then a bipartite graph $G' = (A' \cup B', E')$ is a *complete bipartite subgraph* of G, if $A' \subseteq A$, $B' \subseteq B$ and for all $a \in A'$ and $b \in B'$ we have $(a, b) \in E'$. The graph is called *complete*, since we cannot add any additional edge to it. Given n complete bipartite subgraphs $G_i = (A_i \cup B_i, E_i)$ of G, $i = 1, \ldots, n$, we will say that they *cover* G, if their union is exactly G. More formally the G_i cover G, if we have $\bigcup_{i=1}^{n} A_i = A$, $\bigcup_{i=1}^{n} B_i = B$ and $\bigcup_{i=1}^{n} E_i = E$.

With this one-to-one relationship between bipartite graphs and binary matrices and our definition of complete bipartite subgraphs, we can characterise the Boolean rank as a problem of covering bipartite graphs with complete bipartite subgraphs.

Theorem 3.2 ([Monson et al., 1995])**.** *Let $\boldsymbol{B} \in \{0,1\}^{m \times n}$ be a binary matrix and let $G_{\boldsymbol{B}}$ be its corresponding bipartite graph. Then the following two statements are equivalent:*

1. *$\boldsymbol{B}$ has Boolean rank k, i.e.* $\operatorname{brank}(\boldsymbol{B}) = k$.
2. *The minimum number of complete bipartite subgraphs needed to cover all edges of $G_{\boldsymbol{B}}$ is k.*

We will call the problem from part 2 of Theorem 3.2 the *CCB problem* (short for *covering complete bipartite graphs*).

3.2 Computational Complexity

The Boolean rank of a binary matrix is hard to compute exactly and even hard to approximate. Most of the results were expressed for the CCB problem from Section 3.1, but due the equivalence of the CCB problem and the Boolean rank they also immediately apply to the latter.

From the classic work of Garey and Johnson [1979, problem GT18] it is known that it is NP-hard to solve the CCB problem exactly. Furthermore, Simon [1990] proves that in polynomial time this problem cannot be approximated better than within a factor of 2, unless P = NP.

More recently, Chalermsook et al. [2014] proved that it is hard to compute the Boolean rank of $n \times n$ matrices within a factor of $\frac{n}{2^{\log^{7/8+\varepsilon} n}}$, unless for all $\varepsilon > 0$ the complexity class NP has bounded-error randomised quasi-polynomial time algorithms.

Chapter 4

Sign Rank

This chapter will introduce another rank for binary matrices, which is different from the Boolean rank that we encountered in Chapter 3. This rank is called *sign rank* and it merges techniques from the Boolean rank and from the standard rank. While it is defined on the domain of binary matrices like the Boolean rank, it uses real-valued matrices to represent the rank of the binary matrices.

We will start by formally defining the sign rank in Section 4.1. Section 4.2 will give two equivalent formulations as geometrical problems. Section 4.3 will summarise results on its computational complexity and the following Section 4.4 will provide upper bounds and lower bounds. In Section 4.5 we will see how sign rank has found applications in the field of communication complexity. After that we will see that the sign rank is closely related to the VC dimension in Section 4.6 and we will conclude this chapter by looking at applications of sign rank in data mining in Section 4.7.

4.1 Definition

This section will introduce the *sign rank* of sign matrices and of binary matrices. By *sign matrices* we mean matrices, in which each entry is from the set $\{-, 0, +\}$. A sign matrix will be called *proper*, if it contains no zero entries. We will see that the sign rank basically merges the standard rank from Chapter 2 and the Boolean rank from Chapter 3.

In order to formally define the sign rank, let us introduce the *sign function* which assigns to each real number its sign, i.e.

$$\text{sign} : \mathbb{R} \to \{-, 0, +\}, \qquad z \mapsto \begin{cases} +, & \text{if } z > 0, \\ 0, & \text{if } z = 0, \\ -, & \text{if } z < 0. \end{cases}$$

For a real-valued $m \times n$ matrix $\boldsymbol{A}$ we will write $\text{sign}(\boldsymbol{A})$ to denote the $m \times n$ sign matrix with $(\text{sign}(\boldsymbol{A}))_{ij} = \text{sign}(\boldsymbol{A}_{ij})$ for all $i = 1, \ldots, m$ and $j = 1, \ldots, m$.

Now we can define the sign rank of a sign matrix as the smallest rank of any real-valued matrix, which in each entry has the same sign as the given matrix.

Definition 4.1 ([Alon et al., 2014])**.** *We define the* sign rank *of a proper sign matrix* $\boldsymbol{B} \in \{-, +\}^{m \times n}$ *as the smallest* $k \in \mathbb{N}$*, such that there exists a real-valued matrix* $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ *with* $\text{rank}(\boldsymbol{A}) = k$ *and* $\text{sign}(\boldsymbol{A}_{ij}) = \boldsymbol{B}_{ij}$ *for all* $i = 1, \ldots, m$ *and* $j = 1, \ldots, n$*. We will denote the sign rank of* $\boldsymbol{B}$ *as* $\text{sign-rank}(\boldsymbol{B})$*.*

Notice that the sign rank is only defined for *proper* sign matrices, i.e. for sign matrices without any zero-entries. Thus, in the rest of the thesis we will always assume the sign matrices to be proper when we consider their sign ranks.

In the latter we will also want to compute the sign rank of binary matrices. To this extent let $\boldsymbol{B} \in \{0, 1\}^{m \times n}$ be a binary matrix. Now we map $\boldsymbol{B}$ to a proper sign matrix $\boldsymbol{B}^{\pm} \in \{-, +\}^{m \times n}$ by setting

$$\left(\boldsymbol{B}^{\pm}\right)_{ij} = \begin{cases} +, & \text{if } \boldsymbol{B}_{ij} = 1, \\ -, & \text{if } \boldsymbol{B}_{ij} = 0, \end{cases}$$

for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$. We will call $\boldsymbol{B}^{\pm}$ the *sign matrix of* $\boldsymbol{B}$. Now we define the *sign rank of the binary matrix* $\boldsymbol{B}$ as $\text{sign-rank}(\boldsymbol{B}) = \text{sign-rank}(\boldsymbol{B}^{\pm})$. Notice that with the same construction we can derive a binary matrix from a given proper sign matrix. Thus, in the following we will be using both notions equivalently.

With this understanding we observe that just like the Boolean rank, the domain of sign rank are binary matrices, but to compute the sign rank we will need to argue about the standard rank of real-valued matrices. Thus, one may look at at the sign rank as merging both techniques.

4.2 Relationship to Hyperplane Arrangements

It turns out that sign rank is equivalent to a geometrical problem of linearly separating points in the lowest possible dimensionality, which in turn is equivalent to a problem of hyperplane arrangements. This section is devoted to proving these two claims and to introducing hyperplane arrangements.

We start by introducing the necessary theory of hyperplane arrangements. After that we prove the above mentioned characterisations of sign rank.

Introduction to Hyperplane Arrangements

The theory we introduce in this section is based on the book of Ziegler [1995, chapters 6 and 7]. It will make the reader familiar with hyperplanes and how they dissect the space into multiple regions. For this purpose we will encounter hyperplane arrangements and sign vectors.

Let us start by considering a non-zero vector $\boldsymbol{c} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ and a real number $b \in \mathbb{R}$. Then the set $H = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{x}, \boldsymbol{c} \rangle = b\}$ is called a *hyperplane* if $b = 0$ and for $b \neq 0$ it is called an *affine hyperplane*. We will call the vector $\boldsymbol{c}$ the *normal vector* of H and the number b will be denoted as the hyperplane's *offset from the origin*.

A hyperplane $H = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{x}, \boldsymbol{c} \rangle = 0\}$ with $\boldsymbol{c} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ dissects the space into three different parts: The first set is given by H itself, i.e. all points that lie on the hyperplane. Two more convex open sets are given by $H^+ = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{x}, \boldsymbol{c} \rangle > 0\}$ and by $H^- = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{x}, \boldsymbol{c} \rangle < 0\}$. Notice that by definition all three sets are disjoint and that their union is the whole space $\mathbb{R}^d$. Thus, for each $\boldsymbol{x} \in \mathbb{R}^d$ we have either $\boldsymbol{x} \in H^+$ or $\boldsymbol{x} \in H^-$ or $\boldsymbol{x} \in H$.

The situation that was described in the previous paragraph is visualised in Figure 4.1.

Let us further consider the case that we have more than a single hyperplane; so let $H_1, \ldots, H_n$ be hyperplanes in $\mathbb{R}^d$. Then the set $\mathcal{A} = \{H_1, \ldots, H_n\}$ of these hyperplanes is called a *(linear) hyperplane arrangement in* $\mathbb{R}^d$.

If we are given multiple hyperplanes, then in general the space will be dissected into many *regions*: This can be seen by considering a hyperplane arrangement $\mathcal{A}$ as above and looking at the set $X = \mathbb{R}^d \setminus \bigcup_{i=1}^n H_i$. Observe that X can be seen as the union of multiple open and convex sets; we will call these sets the *regions* of the hyperplane arrangement $\mathcal{A}$. In general, one can prove that n affine hyperplanes can dissect $\mathbb{R}^d$ into at most $\sum_{i=0}^d \binom{n}{i} = O(n^d)$ different regions (see Edelsbrunner [1987, page 7]).

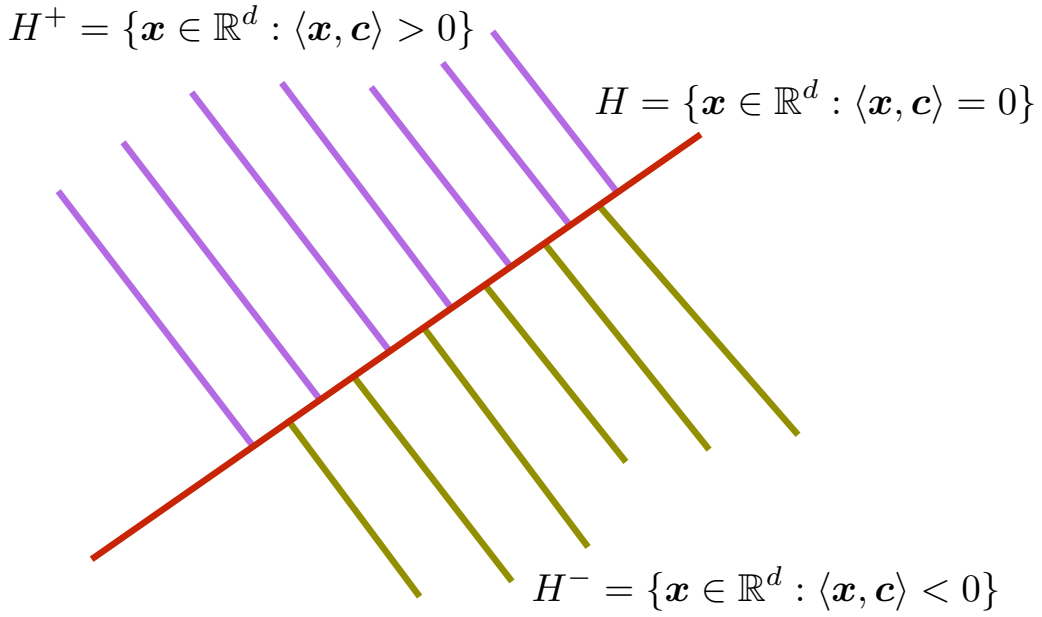

FIGURE 4.1: A single hyperplane H together with the two open convex sets H^+ and H^- into which the space gets dissected.

Figure 4.2 shows three hyperplanes, all of them meeting in the origin and dissecting the space into six regions.

Given multiple hyperplanes, which cut the space into many different regions, we will want to identify the different regions of the dissection. One way to do this is using the so called *sign vectors*.

Definition 4.2. *Let $\mathcal{A} = \{H_1, \ldots, H_n\}$ be a hyperplane arrangement in $\mathbb{R}^d$. Then $t \in \{-, 0, +\}^n$ is the* sign vector *of a point $\boldsymbol{x} \in \mathbb{R}^d$, if for all components $i = 1, \ldots, n$, we have:*

$$t_i = \begin{cases} +, & \text{if } \boldsymbol{x} \in H_i^+, \\ 0, & \text{if } \boldsymbol{x} \in H_i, \\ -, & \text{if } \boldsymbol{x} \in H_i^-. \end{cases}$$

Notice that within a region of a hyperplane arrangement $\mathcal{A}$ all points will have exactly the same sign vector. Further observe that if two points are in different regions, then they must have different sign vectors. Thus, there is a one-to-one relationship between regions and sign vectors.

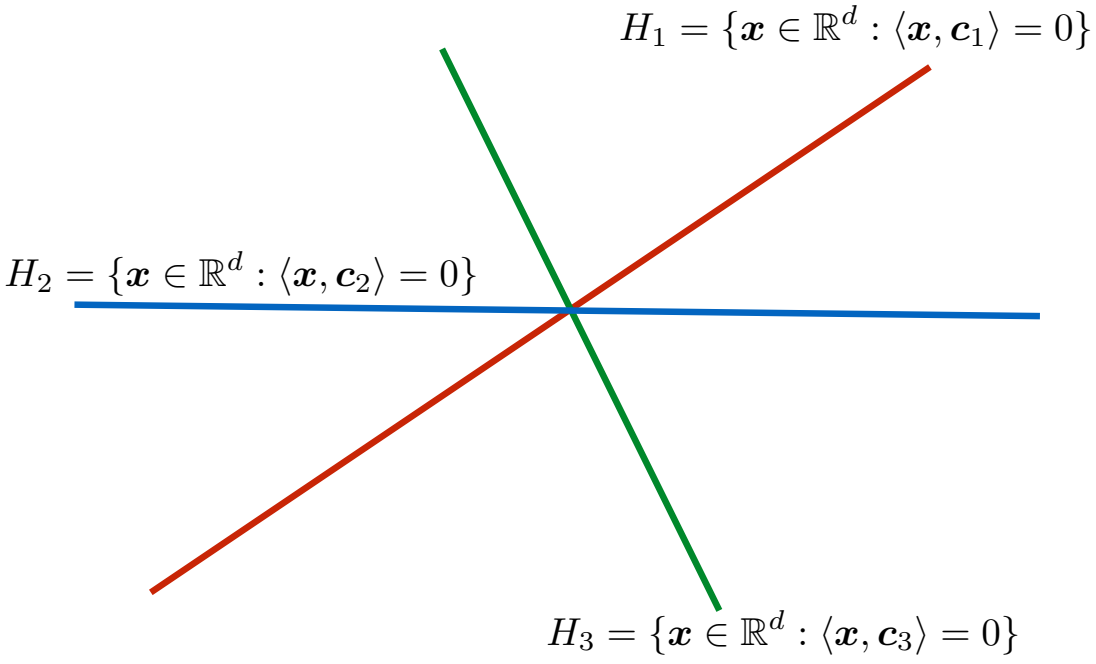

FIGURE 4.2: Three hyperplanes in $\mathbb{R}^2$.

An example of a hyperplane arrangement in the plane consisting of three hyperplanes together with the sign vectors of the regions of the arrangement can be found in Figure 4.3. Notice that the colours of the components of the sign vectors indicate from which hyperplane they originate.

Now we will characterise the sign vector of a point using the sign function. Assume we are given a hyperplane arrangement of $\mathcal{A}$ of n hyperplanes $H_1, \ldots, H_n$ in $\mathbb{R}^d$, together with their normal vectors $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n$. Also, let $\boldsymbol{x} \in \mathbb{R}^d$ and let t be the sign vector of x with respect to $\mathcal{A}$. Then it is not hard to see that we have the following equality:

$$t = (\text{sign}(\langle \boldsymbol{x}, \boldsymbol{c}_1 \rangle), \ldots, \text{sign}(\langle \boldsymbol{x}, \boldsymbol{c}_n \rangle)) \,. \tag{4.1}$$

Looking at more than a single sign vector, we define the *set of sign vectors*, $\mathcal{V}(\mathcal{A})$, of a hyperplane arrangement $\mathcal{A} = \{H_1, \ldots, H_n\}$ by setting

$$\mathcal{V}(\mathcal{A}) = \left\{ t \in \{-, 0, +\}^n : t \text{ is the sign vector for some } \boldsymbol{x} \in \mathbb{R}^d \right\}.$$

Notice that we get an equivalence relationship on the set of all hyperplane arrangements by calling two hyperplane arrangements *equivalent* if their sets of sign vectors agree.

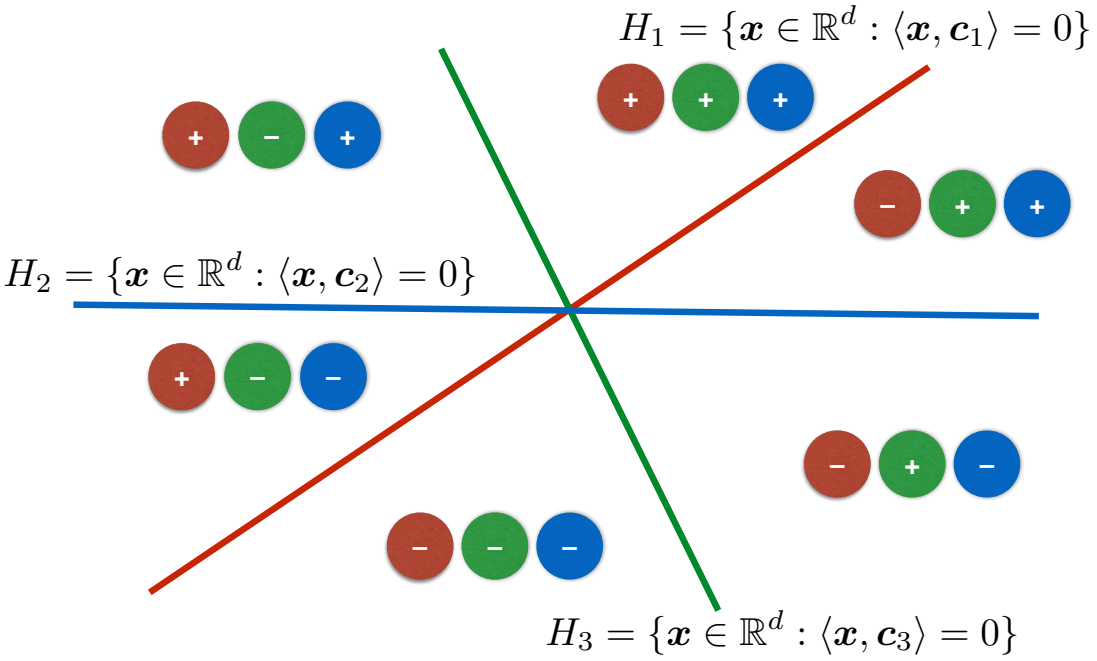

FIGURE 4.3: A hyperplane arrangement in $\mathbb{R}^2$ with three hyperplanes together with the sign vectors of the regions.

In the rest of the thesis we will not consider the case that sign vectors contain zeros, i.e. we will only consider points which do not lie on any hyperplane and which instead are in the open regions of the hyperplane arrangement.

By now we have introduced enough theory about hyperplane arrangements that is required for this thesis. Nonetheless, the author wants to point out that the theory of hyperplane arrangements is much more versatile than what we have seen here. For example, one can proof that hyperplane arrangements, oriented matroids and zonotopes are basically equivalent. For more details on hyperplane arrangements, the reader is referred to Ziegler [1995], Orlik and Terao [1992] and Stanley [2004]. A good entry point to the work done on oriented matroids is Ziegler [2002].

Sign Rank and Hyperplane Arrangements are Equivalent

With all the previous definitions we can prove the main result of this section, which relates sign rank to the theory of hyperplane arrangements and a geometrical problem of embedding linearly separable points in low-dimensional spaces.

The following theorem is an adaptation of the results of Paturi and Simon [1984]. Their findings were slightly altered to fit more directly into the framework of hyperplane arrangements.

Theorem 4.3. *Let $\boldsymbol{B} \in \{0,1\}^{m\times n}$ be a binary matrix and let $\boldsymbol{B}^{\pm} \in \{-,+\}^{m\times n}$ be the sign matrix of $\boldsymbol{B}$. Also, let $d \in \mathbb{N}$. Then the following statements are equivalent:*

1. *$\boldsymbol{B}$ has sign rank at most d, i.e.* sign-rank$(\boldsymbol{B}) \leq d$.
2. *$\boldsymbol{B}^{\pm}$ has sign rank at most d, i.e.* sign-rank$(\boldsymbol{B}^{\pm}) \leq d$.
3. *There exist points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, such that for all $j = 1, \ldots, n$, the classes $C_j = \{\boldsymbol{x}_i : \boldsymbol{B}^{\pm}_{ij} = +\}$ and $\bar{C}_j = \{\boldsymbol{x}_i : \boldsymbol{B}^{\pm}_{ij} = -\}$ are strictly linearly separable with hyperplanes through the origin*[1].
4. *There exists a hyperplane arrangement $\mathcal{A} = \{H_1, \ldots, H_n\}$ in $\mathbb{R}^d$ with $\boldsymbol{B}^{\pm} \subseteq \mathcal{V}(\mathcal{A})$, i.e. the rows of $\boldsymbol{B}^{\pm}$ are a subset of the set sign vectors of $\mathcal{A}$.*

Proof. $1 \Leftrightarrow 2$: This follows immediately from the definition of sign rank sign matrices and binary matrices.

$2 \Rightarrow 3$: Let $\boldsymbol{A} \in \mathbb{R}^{m\times d}$ be a matrix with $\operatorname{sign}(\boldsymbol{A}) = \boldsymbol{B}^{\pm}$ and $\operatorname{rank}(\boldsymbol{A}) \leq d$, which exists by the claim of point 2 from the theorem. Now let $\boldsymbol{L} \in \mathbb{R}^{m\times d}$ and $\boldsymbol{R} \in \mathbb{R}^{n\times d}$ be a decomposition of $\boldsymbol{A}$ with $\boldsymbol{L}\boldsymbol{R}^T = \boldsymbol{A}$.

Consider the rows $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ of $\boldsymbol{L}$ as points in $\mathbb{R}^d$ and the rows $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n$ of $\boldsymbol{R}$ as normal vectors of hyperplanes. For all i and j, we have $\operatorname{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle) = \operatorname{sign}(\boldsymbol{A}_{ij}) = \boldsymbol{B}^{\pm}_{ij}$. Thus, fixing some $\boldsymbol{r}_j$, we see that the points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ get strictly linearly separated into two classes

$$
\begin{aligned}
C_j &= \{\boldsymbol{l}_i : \langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle > 0\} = \{\boldsymbol{l}_i : \operatorname{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle) = +\} = \{\boldsymbol{l}_i : \boldsymbol{B}^{\pm}_{ij} = +\}, \\
\bar{C}_j &= \{\boldsymbol{l}_i : \langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle < 0\} = \{\boldsymbol{l}_i : \operatorname{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle) = -\} = \{\boldsymbol{l}_i : \boldsymbol{B}^{\pm}_{ij} = -\}.
\end{aligned}
$$

Since this is the case for all $j = 1, \ldots, n$, this proves that in $\mathbb{R}^d$ points as required by the third part of the theorem exist.

$3 \Rightarrow 2$: Let the points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ be as given in the theorem and for all $j = 1, \ldots, n$ denote the normal vector of the hyperplane separating the classes C_j and $\bar{C}_j$ by $\boldsymbol{r}_j \in \mathbb{R}^d$. Without loss of generality we assume that $\boldsymbol{x}_i \in C_j$ iff $\langle \boldsymbol{x}_i, \boldsymbol{r}_j \rangle > 0$ (otherwise swap the sign of $\boldsymbol{r}_j$).

[1] See Definition A.1 for a definition of strict linear separability.

Using the strict linear separability of the C_j and $\bar{C}_j$, we observe that $\langle \boldsymbol{x}_i, \boldsymbol{r}_j \rangle > 0$ iff $\boldsymbol{x}_i \in C_j$ iff $\boldsymbol{B}^{\pm}_{ij} = +$ and $\langle \boldsymbol{x}_i, \boldsymbol{r}_j \rangle < 0$ iff $\boldsymbol{x}_i \in \bar{C}_j$ iff $\boldsymbol{B}^{\pm}_{ij} = -$. Thus, we have $\text{sign}(\langle \boldsymbol{x}_i, \boldsymbol{r}_j \rangle) = \boldsymbol{B}^{\pm}_{ij}$ for all entries of the matrix $\boldsymbol{B}^{\pm}$.

Now we write the $\boldsymbol{x}_i$ into the rows of a matrix $\boldsymbol{L} \in \mathbb{R}^{m \times d}$ and the $\boldsymbol{r}_j$ into the rows of a matrix $\boldsymbol{R} \in \mathbb{R}^{n \times d}$. Then setting $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{R}^T$, we obtain $\text{sign}(\boldsymbol{A}) = \text{sign}(\boldsymbol{L}\boldsymbol{R}) = \boldsymbol{B}^{\pm}$ and therefore $\text{sign-rank}(\boldsymbol{B}^{\pm}) \leq d$.

$2 \Rightarrow 4$: Let $\boldsymbol{A}$ be a real-valued $m \times n$ matrix with $\text{sign}(\boldsymbol{A}) = \boldsymbol{B}^{\pm}$ and $\text{rank}(\boldsymbol{A}) \leq d$. Let $\boldsymbol{L} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{R} \in \mathbb{R}^{n \times d}$ be a decomposition of $\boldsymbol{A}$ with $\boldsymbol{L}\boldsymbol{R}^T = \boldsymbol{A}$. We denote the rows of $\boldsymbol{L}$ by $\boldsymbol{l}_i$ and the rows of $\boldsymbol{R}$ by $\boldsymbol{r}_j$.

Now consider the hyperplane arrangement $\mathcal{A} = \{H_1, \ldots, H_n\}$, where H_j is the hyperplane given by the normal vector $\boldsymbol{r}_j$. We need to show $\boldsymbol{B}^{\pm} \subseteq \mathcal{V}(\mathcal{A})$.

Denote the i'th row of $\boldsymbol{B}^{\pm}$ by $\boldsymbol{b}^{\pm}_i$. Then by point two of the theorem we get that for all i and j, we have $\text{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle) = \boldsymbol{B}^{\pm}_{ij}$. Thus the point $\boldsymbol{l}_i$ has sign vector $\boldsymbol{b}^{\pm}_i$, since using the characterisation of sign vectors from Equation 4.1 we obtain

$$(\text{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_1 \rangle), \ldots, \text{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_n \rangle)) = \boldsymbol{b}^{\pm}_i.$$

Therefore, for each row of $\boldsymbol{B}^{\pm}$ there exists a point with this particular sign vector and thus $\boldsymbol{B}^{\pm} \subseteq \mathcal{V}(\mathcal{A})$.

$4 \Rightarrow 2$: Let the hyperplane arrangement $\mathcal{A} = \{H_1, \ldots, H_n\}$ be as stated in the theorem. We will denote the normal vectors of the H_i by $\boldsymbol{c}_i$ and we will denote the i'th row of $\boldsymbol{B}^{\pm}$ by $\boldsymbol{b}^{\pm}_i$.

For each $i = 1, \ldots, m$, we can pick a point $\boldsymbol{v}_i \in \mathbb{R}^d$ that has sign vector $\boldsymbol{b}^{\pm}_i$ with respect to $\mathcal{A}$ (by the claim from point 4 of the theorem). Now observe that for all i and j, we have $\text{sign}(\langle \boldsymbol{v}_i, \boldsymbol{c}_j \rangle) = +$ iff $\boldsymbol{B}^{\pm}_{ij} = +$ and thus $\langle \boldsymbol{v}_i, \boldsymbol{c}_j \rangle > 0$ iff $\boldsymbol{B}^{\pm}_{ij} = +$. Similarly, we obtain $\langle \boldsymbol{v}_i, \boldsymbol{c}_j \rangle < 0$ iff $\boldsymbol{B}^{\pm}_{ij} = -$.

Thus, writing the $\boldsymbol{v}_i$ into the rows of a matrix $\boldsymbol{L} \in \mathbb{R}^{m \times d}$ and the $\boldsymbol{c}_j$ into the rows of a matrix $\boldsymbol{R} \in \mathbb{R}^{n \times d}$, we have $\text{sign}(\boldsymbol{L}\boldsymbol{R}^T) = \boldsymbol{B}^{\pm}$ and furthermore $\text{rank}(\boldsymbol{L}\boldsymbol{R}^T) \leq d$. This proves $\text{sign-rank}(\boldsymbol{B}^{\pm}) \leq d$. □

From this theorem it follows that if one can minimize either of the four equivalent statements, this will also minimize the other ones. For example, if we can find a hyperplane arrangement of minimum dimensionality, such that it contains the sign vectors of a matrix $\boldsymbol{B}^{\pm}$, then we know the exact sign rank of $\boldsymbol{B}^{\pm}$.

The following corollary gives a trivial upper bound on the sign rank using the characterisation with hyperplane arrangements.

Corollary 4.4. *Let $\boldsymbol{B}^{\pm} \in \{-,+\}^{m\times n}$ be a sign matrix. Then the sign rank of $\boldsymbol{B}^{\pm}$ is bounded from above by* $\min\{m,n\}$*, i.e.* sign-rank$(\boldsymbol{B}^{\pm}) \leq \min\{m,n\}$.

Proof. Without loss of generality, we may assume that $m \geq n$ (otherwise we can transpose $\boldsymbol{B}^{\pm}$). Now consider the hyperplane arrangement $\mathcal{A}$ consisting of the standard basis of $\mathbb{R}^n$. Observe that each vector $\boldsymbol{x} \in \{-1,1\}^n$ has sign vector $t = \text{sign}(\boldsymbol{x})$. Thus, we have $\mathcal{V}(\mathcal{A}) = \{-,+\}^n$ and therefore also $\boldsymbol{B}^{\pm} \subseteq \{+,-\}^n = \mathcal{V}(\mathcal{A})$. Applying point 4 of Theorem 4.3 we get that sign-rank$(\boldsymbol{B}) \leq n$. □

Of course, this result is not very surprising since when using matrix factorisations, for a binary matrix $\boldsymbol{B}$ we get sign-rank$(\boldsymbol{B}) \leq$ rank$(\boldsymbol{B}) \leq \min\{m,n\}$, as we will see in Chapter 6. Nonetheless, we did not use this argument in the proof of the corollary. Our approach only used geometry and is somewhat more intuitive and clearly more constructive than the one using standard rank as an upper bound.

4.3 Computational Complexity and Algorithms

The computational complexity of sign rank is an open problem and is closely related to other well-known problems. This section will present what is known about the computational complexity of sign rank.

Let us denote the problem of deciding whether the sign rank of a given matrix is at most k by k-SIGNRANK, i.e. k-SIGNRANK $= \{\boldsymbol{B} : \text{sign-rank}(\boldsymbol{B}) \leq k\}$. Lately, both Basri et al. [2009] and Bhangale and Kopparty [2015] independently proved that 3-SIGNRANK is NP-hard. The work of Kang and Müller [2012] implies that k-SIGNRANK is NP-hard for all $k \geq 3$. For their proofs, all of them used reductions to a result of Shor [1991], which shows that deciding whether an arrangement of pseudolines is stretchable is NP-hard.

A result of Canny [1988] implies that sign rank is in PSPACE.

Furthermore, the results of Mnev [1985] and Mnev [1988] imply that k-SIGNRANK is polynomial-time equivalent to the existential theory of the reals. The *existential theory of the reals (ETR)* is the problem of deciding whether a set of polynomial equalites, inequalites and strict inequalities with integer coefficients has a real-valued solution. This problem is known to be NP-hard and to be in PSPACE, while it is not known whether ETR is a member in NP or not. Thus, according to Kang and Müller [2012] proving k-SIGNRANK to be a member of NP would be a minor breakthrough in computational

complexity theory. A list of other problems, that are polynomial-time equivalent to ETR can be found in the work of Matousek [2014].

Moreover, Kang and Müller [2012] gave examples of binary matrices for which all real-valued matrices achieving the sign rank must have exponential size in the input. More formally, they constructed a family of $n \times n$ sign matrices $\boldsymbol{B}_n$ with sign-rank$(\boldsymbol{B}_n) = 3$, such that for all real-valued matrices $\boldsymbol{A}$ with $\text{sign}(\boldsymbol{A}) = \boldsymbol{B}_n$ and $\text{rank}(\boldsymbol{A}) = 3$, $\boldsymbol{A}$ must have entries with size exponential in n. This result rules out proving that sign rank is a member of NP by nondeterministically guessing a real-valued matrix, that achieves the sign rank. Of course, this only applies if we store each entry of the matrix bit by bit. It may still be possible to store the matrix in a more "economic" way.

Let us finish this section by mentioning existing algorithms for sign rank. A polynomial-time algorithm for determining whether a sign matrix has sign rank at most two was given by Bhangale and Kopparty [2015]. Alon et al. [2014] gave an approximation algorithm for $n \times n$ sign matrices with approximation factor $\frac{n}{\log n}$.

4.4 Upper Bounds and Lower Bounds

In this section we will summarise existing results on upper and lower bounds for sign matrices. We will be particularly interested in how large the sign rank of sign matrices may get. We will also see a lower bound for the sign rank of given sign matrices.

Let $d(m, n)$ denote the highest sign rank of all $m \times n$ sign matrices, i.e.

$$d(m, n) = \max\{\text{sign-rank}(\boldsymbol{B}) : \boldsymbol{B} \in \{+, -\}^{m \times n}\}.$$

The trivial upper bound is given by $d(m, n) \leq \min\{m, n\}$, as we had seen in Corollary 4.4. The trivial lower bound is $d(m, n) \geq \log(\min\{m, n\})$, because a hyperplane arrangement with $\min\{m, n\}$ hyperplanes in $\mathbb{R}^d$ can intersect the space into at most $O(\min\{m, n\}^d)$ regions as we had seen in Section 4.2.

Non-trivial bounds were derived by Alon et al. [1985]. Their result shows that there exist matrices sign rank linear in the size of the matrix. In particular, they proved that for $n \times n$ sign matrices, we have

$$n/32 \leq d(n, n) \leq \left(\frac{1}{2} + o(1)\right) n.$$

This result was derived using a counting argument. For a long time it was an open problem to prove for a particular matrix that it has large sign rank. We will see an

example of such a matrix in Section 5.5. The lower bound on $d(n, n)$ also shows that reconstructing the sign pattern of a real-valued matrix can be almost as difficult as reconstructing it exactly.

Given a sign matrix as an input, Forster [2002] was able to give a lower bound on its sign rank. Forster's theorem considers sign matrices with entries from the set $\{-1, 1\}$ instead of $\{-, +\}$ and uses the spectrum of the matrix to bound the sign rank. To give his result we will need the *operator norm* $||\cdot||$ of a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, which is defined by

$$||\boldsymbol{A}|| = \sup\{||\boldsymbol{A}\boldsymbol{x}|| : \boldsymbol{x} \in \mathbb{R}^n, ||\boldsymbol{x}||_2 \leq 1\}.$$

It is known (see, e.g. Boyd and Vandenberghe [2004, page 636]) that this operator norm is the same as the largest singular value of $\boldsymbol{A}$, i.e. $||\boldsymbol{A}|| = \sigma_1(\boldsymbol{A})$.

Theorem 4.5 ([Forster, 2002])**.** *Let $\boldsymbol{B} \in \{-1, 1\}^{m\times n}$. Then we get*

$$\text{sign-rank}(\boldsymbol{B}) \geq \frac{\sqrt{mn}}{||\boldsymbol{B}||}.$$

This result was improved by Forster and Simon [2006] by using the whole spectrum of the the sign matrix (instead of just the largest singular value).

Theorem 4.6 ([Forster and Simon, 2006])**.** *Let $\boldsymbol{B} \in \{-1, +1\}^{m\times n}$. Let $r = \text{rank}(\boldsymbol{B})$ and let $\sigma_1(\boldsymbol{B}) \geq \cdots \geq \sigma_r(\boldsymbol{B}) > 0$ be the singular values of $\boldsymbol{B}$. Denote the sign rank of $\boldsymbol{B}$ by d. Then*

$$d\sum_{i=1}^{d} \sigma_i^2(\boldsymbol{B}) \geq mn.$$

It is also interesting to consider what sign rank a random sign matrix will have with high probability. For example, in the real-valued domain it is well-known that if we sample an $n \times n$ matrix uniformly at random from the set $[0, 1]^{n\times n}$, then it will have full rank with probability one (from the proof of Theorem 2.20 in Rudin [1987]). For sign matrices Paturi and Simon [1984] were able to prove a similar result. They proved a lower bound on the sign ranks of *almost all* sign matrices, which shows that a random $n \times n$ sign matrix will have sign rank $\omega(n^{1/2-\varepsilon})$ for all $\varepsilon \in (0, 1/2)$.

In order to state the theorem, we will need a further definition to clarify the term *almost all*. Let $(\mathbb{B}_n)_{n\in\mathbb{N}}$ denote the family of sets of all proper $n \times n$ sign matrices, i.e. for all n we have $\mathbb{B}_n = \{-1, 1\}^{n\times n}$. Then a *property* P of $(\mathbb{B}_n)_{n\in\mathbb{N}}$ is a family of subsets $P = (P_n)_{n\in\mathbb{N}}$, such that the P_n satisfy $P_n \subseteq \mathbb{B}_n$ for all n. We say that P holds for *almost all* proper sign matrices, if for all $\varepsilon > 0$ there exists an $n_0 \in \mathbb{N}$, such that for all $n > n_0$

we have

$$\frac{|P_n|}{|\mathbb{B}_n|} > 1 - \varepsilon.$$

Notice that we could as well write $|P_n|/|\mathbb{B}_n| \to 1$ for $n \to \infty$.

For example, let $P = (P_n)_{n\in\mathbb{N}}$ denote the set of all proper $n \times n$ sign matrices with at least two one-entries. Then a quick computations shows that P holds true for almost all proper sign matrices.

Now we can write down the theorem, which shows that random sign matrices will have a large sign rank with high probability. In the theorem the properties are the sets $P_n = \left\{\boldsymbol{B} \in \{-1,1\}^{2^n\times 2^n} : \text{sign-rank}(\boldsymbol{B}) \geq 2^{n/2-\log(n/2)}\right\}$.

Theorem 4.7 ([Paturi and Simon, 1984])**.** *Almost all proper $2^n \times 2^n$ sign matrices $\boldsymbol{B} \in \{-1,1\}^{2^n\times 2^n}$ satisfy the inequality*

$$\text{sign-rank}(\boldsymbol{B}) \geq 2^{n/2-\log(n/2)}.$$

Using Theorem 4.5 and Theorem 4.9, Forster [2002] was able to improve the previous result by making it depend on the operator norm of the random sign matrix.

Theorem 4.8 ([Forster, 2002])**.** *For almost all proper sign matrices $\boldsymbol{B} \in \{-1,1\}^{2^n\times 2^n}$,*

$$\text{sign-rank}(\boldsymbol{B}) \geq 2^{n-\log||\boldsymbol{B}||}.$$

4.5 Applications in Communication Complexity

This section will define *probabilistic communication complexity*, which is also known as *unbounded-error communication complexity*. It was first introduced in 1984 by the paper of Paturi and Simon [1984]. Their article proved that probabilistic communication complexity and sign rank are essentially equivalent.

In unbounded-error communication complexity we assume to have two parties, Alice and Bob. Both have access to an infinite number of private random bits and they also have infinite computational power. Now Alice receives some information $x \in \{0,1\}^n$ and Bob receives information $y \in \{0,1\}^n$. Together they want to compute a function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with a minimal amount of communication. In order to compute this function, they are sending messages in turns. Each party randomly decides which message it should send by using its private random bits [Razborov and Sherstov, 2008].

We say that their protocol *computes the function* f if on every input (x, y), the output is correct with probability at least $1/2$. The *cost of a protocol* is the worst-case number of bits Alice and Bob exchange over all inputs (x, y). Now the *unbounded-error communication complexity of the function* f is the least cost of any protocol computing f. We will denote it by $\tilde{C}_f$ [Razborov and Sherstov, 2008].

To relate the just introduced probabilistic communication complexity with sign rank, we will see that there exists a one-to-one relationship between binary functions mapping from the set $\{0,1\}^n \times \{0,1\}^n$ to $\{0,1\}$ and binary matrices: Given f as above, we define the *binary matrix of* f, denoted by $\boldsymbol{M}_f$, as the $2^n \times 2^n$ sign matrix with $(\boldsymbol{M}_f)_{ij} = f(\mathrm{bin}_n(i), \mathrm{bin}_n(j))$ for all $i = 1, \ldots, 2^n$ and $j = 1, \ldots, 2^n$. Here $\mathrm{bin}_n : \{0, \ldots, 2^n - 1\} \to \{0,1\}^n$ gives the binary encoding of a given natural number. Notice that this construction is bijective.

Using this bijection between binary functions and binary matrices, it was proven by Paturi and Simon [1984] that the unbounded-error communication complexity of a function f is essentially equivalent to the logarithm of the sign rank of the binary matrix $\boldsymbol{M}_f$.

Theorem 4.9 ([Paturi and Simon, 1984])**.** *Let* $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *be a function and let* $\boldsymbol{M}_f$ *be the matrix of* f*. Then*

$$\lceil \log(\text{sign-rank}(\boldsymbol{M}_f)) \rceil \leq \tilde{C}_f \leq \lceil \log(\text{sign-rank}(\boldsymbol{M}_f)) \rceil + 1.$$

This is a highly interesting result and launched a lot of related research, that we will present in the following sections of this chapter. Since the theorem proves that all results that apply to sign rank will immediately apply to unbounded-error communication complexity, all lower and upper bounds that we encountered in Section 4.4 will immediately serve as bounds on the communication complexity of the associated functions.

While the standard probabilistic communication complexity takes into account protocols that are correct with probability at least $1/2$, Krause [1991] also looks into protocols that compute a function f with probability at least $1/2 - 1/s$ for $s \in \mathbb{N}$. His results prove that under these conditions the unbounded-error communication complexity of every function f is at least $\frac{1}{4}(n - \log(||\boldsymbol{M}_f||) - \log(s) - 2)$, where $||\cdot||$ again denotes the operator norm and $\boldsymbol{M}_f$ is the binary matrix of f.

Sign rank also turned out to be useful for the separation of complexity classes from communication complexity. For more information on this topic the reader is referred to Razborov and Sherstov [2008] and Forster et al. [2001] and references therein.

4.6 Applications in Learning Theory

This section will discuss how sign rank has found applications in learning theory. We will see that sign rank is related to the Vapnik-Chervonenkis (VC) dimension and that sign rank can be slightly altered in order to theoretically analyse how powerful large margin classifiers are.

We start by introducing the Vapnik-Chervonenkis dimension and discussing how it is related to sign rank. For lack of space we will only introduce the definition of the VC dimension and mostly work with intuition instead of going into full details. A more detailed coverage of the VC dimension can for example be found in the book of Kearns and Vazirani [1994].

We start by defining the VC dimension, which was originally introduced in the seminal paper of Vapnik and Chervonenkis [1971]. Here we present an equivalent definition of Alon et al. [2014] as it requires less notation.

Definition 4.10 (Vapnik-Chervonenkis (VC) dimension, [Alon et al., 2014])**.** *Let $\boldsymbol{B}$ be a proper sign matrix. Then a subset C of the columns of $\boldsymbol{B}$ is called* shattered*, if each of the $2^{|C|}$ different patterns of plusses and minuses appears in some row in the restriction of $\boldsymbol{B}$ to the columns in C. The* Vapnik-Chervonenkis dimension *(or in short* VC dimension*) of $\boldsymbol{B}$ is the maximum size of a shattered subset of columns. We will denote it by* $\mathrm{VC}(\boldsymbol{B})$*.*

In the following example we will compute the VC dimension of a sign matrix in order to better understand this definition.

Example 4.11. *Consider the 8×3 sign matrix $\boldsymbol{B}$ with*

$$\boldsymbol{B} = \begin{pmatrix} + & - & + \\ + & + & + \\ - & - & - \\ - & + & + \\ + & - & - \\ + & + & + \\ - & - & - \\ - & + & + \end{pmatrix}.$$

This matrix $\boldsymbol{B}$ has VC dimension 2: If we pick the first two columns of $\boldsymbol{B}$, then their rows contain all the vectors from the set $\{-,+\}^2$. Thus, the first two columns give us a shattered subset. On the other hand, if we pick all three columns of $\boldsymbol{B}$, then we observe

that the vector $\begin{pmatrix} - & + & - \end{pmatrix}$ *is missing and therefore they are not shattered. This means that* $\mathrm{VC}(\boldsymbol{B}) = 2$.

At this point let us remark a highly important result from learning theory. The seminal paper of Blumer et al. [1989] proved that the VC dimension is equivalent to the PAC learning sample complexity, which was introduced by Valiant [1984]. More details about this can be found in Kearns and Vazirani [1994].

Now let us first look into why the VC dimension and sign rank are interesting for learning theory. For this we will first see how sign matrices resemble multi-class classification problems and then we will motivate how VC dimension and sign rank relate to learnability.

Observe that we can interpret an $m \times n$ sign matrix $\boldsymbol{B}$ as the labels of points in a multi-class binary classification problem: Assume that we have some data points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m$ and n classes $C_1, \dots, C_n$. Then the label of $\boldsymbol{x}_i$ with respect to class C_j is given by $\boldsymbol{B}_{ij} \in \{-, +\}$. Thus, the matrix $\boldsymbol{B}$ tells us the labels of all m points with respect to the n classes. Notice that we can also construct a matrix from a multi-class classification problem.

Now the rough intuition is that one can interpret matrices $\boldsymbol{B}$ (or equivalently multi-class classification problems) with large VC dimension as 'difficult' to classify for all possible machine learning algorithms. Here 'difficult' means that the algorithms would need a large training data set or will always have a large test error. Similarly, matrices with small VC dimension are more 'easily' learnable, i.e. they can be learned with a 'small' sample size. Of course, this is only a very loose matter of speaking and the underlying theory is more complicated.

Sign rank, on the other hand, only allows for linear classifiers (this is implied by Theorem 4.3). Thus, the possible models are more restricted than in the VC dimension.

Comparing VC dimension and sign rank we may interpret that the VC dimension corresponds to what is learnable (as it allows for a large class of classifiers), whereas sign rank corresponds to what is efficiently learnable (as separating points by hyperplanes can be done efficiently in practice).

Given this intuition it is interesting to look into how the VC dimension and the sign rank of a matrix relate to one another quantitatively. First results showed that the VC dimension is a lower bound on the sign rank. In Alon et al. [1987] the authors proved the existence of $n \times n$ sign matrices which have VC dimension two, but sign rank growing to infinity in n. Thus, the gap between 'learnability' and 'learnability with hyperplanes' can grow arbitrarily large.

Alon et al. [2014] further explored this problem and gave bounds on the maximum sign rank a matrix of a given VC dimension may have. Let us introduce the function $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, where $f(n, d)$ gives the maximum sign rank of all $n \times n$ matrices with VC dimension d, i.e.

$$f(n, d) = \max\{\text{sign-rank}(\boldsymbol{B}) : \boldsymbol{B} \in \{-, +\}^{n \times n} \text{ and } \text{VC}(\boldsymbol{B}) = d\}.$$

The results which were derived for lower bounds on $f(n, d)$ are given the following theorem.

Theorem 4.12 ([Alon et al., 2014])**.** *The following lower bounds on $f(n, d)$ hold:*

1. $f(n, 2) \geq \Omega(n^{1/2}/\log n)$.
2. $f(n, 3) \geq \Omega(n^{8/15}/\log n)$.
3. $f(n, 4) \geq \Omega(n^{2/3}/\log n)$.
4. *For every $d > 4$,*

$$f(n, d) \geq \Omega(n^{1-(d^2+5d+2)/(d^3+2d^2+3d)}/\log n).$$

Alon et al. [2014] were also able to obtain an upper bound on $f(n, d)$.

Theorem 4.13 ([Alon et al., 2014])**.** *For every fixed $d \geq 2$,*

$$f(n, d) \leq O(n^{1-1/d}).$$

Concluding the results from the above two theorems, we realise that even for sign matrices with small VC dimensions it may be very hard to perform the classification of the underlying multi-class classification problem with hyperplanes.

We will spend the rest of this section discussing how sign rank can be altered in order to characterise the performance of large margin classification with hyperplanes. Notice that for example when we use support vector machines we are usually not just interested in finding hyperplanes that separate our data, but we want to find hyperplanes that achieve large margins.

This led Forster et al. [2003] and Ben-David et al. [2001] to pose questions closely related to sign rank in order to be able to argue about the margins of the hyperplanes. We will now introduce some notation in order to be able to state some of the results that were obtained in this line of research.

Definition 4.14 ([Forster and Simon, 2006])**.** *A* linear arrangement *representing a matrix* $\boldsymbol{B} \in \{-,+\}^{m\times n}$ *is given by vectors* $\boldsymbol{l}_1,\dots,\boldsymbol{l}_m,\boldsymbol{r}_1,\dots,\boldsymbol{r}_n \in \mathbb{R}^d$ *with Euclidean length* $||\boldsymbol{l}_i||_2 = ||\boldsymbol{r}_j||_2 = 1$*, such that* $\operatorname{sign}(\langle \boldsymbol{l}_i, \boldsymbol{r}_j\rangle) = \boldsymbol{B}_{ij}$ *for all* $i = 1,\dots,m$ *and* $j = 1,\dots,n$*.*

We call the parameter d *the* dimension *of the linear arrangement. The* minimal margin *of the arrangement is given by* $\min_{i,j}\langle \boldsymbol{l}_i, \boldsymbol{r}_j\rangle$*. The* average margin *is defined by* $\frac{1}{mn}\sum_{i,j}\langle u_i, v_j\rangle$*.*

Notice that in the definition the vectors $\boldsymbol{l}_i$ and $\boldsymbol{r}_j$ give a sign rank decomposition of $\boldsymbol{B}$ when we write them into matrices $\boldsymbol{L}$ and $\boldsymbol{R}$, but here we require them to be of unit length (otherwise the margins could be scaled arbitrarily). Also, the definitions allow d to be strictly larger than the sign rank of $\boldsymbol{B}$.

With this notion of margins for linear arrangements there are multiple interesting questions: Firstly, when we just want to minimise the dimensionality d, then we arrive at the sign rank problem. Secondly, it is interesting to maximise the minimal margins and the average margins. Thirdly, it makes sense to think about how the margins behave for different dimensionalities.

Since the first question was already excessively discussed in the previous parts of the chapter, we will only consider the last two ones. To the author's best knowledge the tightest bounds on the minimal margins were derived by Forster et al. [2003]. Their result is given in the following theorem.

Theorem 4.15 ([Forster et al., 2003])**.** *Let* $\boldsymbol{B} \in \{-1,+1\}^{m\times n}$ *be a sign matrix and let* $\tilde{\boldsymbol{M}} \in \mathbb{R}^{m\times n}$ *be a real-valued matrix with* $\operatorname{sign}(\tilde{\boldsymbol{M}}_{ij}) = \boldsymbol{B}_{ij}$ *for all* i *and* j*. Also, let* $\boldsymbol{l}_1,\dots,\boldsymbol{l}_m,\boldsymbol{r}_1,\dots,\boldsymbol{r}_n$ *be a linear arrangement representing* $\boldsymbol{B}$*.*

Then the following upper bound on the minimal margin holds:

$$\min_{i,j} |\langle \boldsymbol{l}_i, \boldsymbol{r}_j\rangle| \leq \frac{\sqrt{m}\cdot||\tilde{\boldsymbol{M}}||}{\sum_j \left(\sum_i |\tilde{\boldsymbol{M}}_{ij}|\right)^2},$$

where $||\cdot||$ *denotes the operator norm of a matrix.*

The following theorem states an upper bound on the average margin.

Theorem 4.16 ([Forster and Simon, 2006])**.** *Let* $\boldsymbol{B}, \boldsymbol{l}_i$ *and* $\boldsymbol{r}_j$ *be as in Theorem 4.15. Then we have the following upper bound on the on the average margin:*

$$\sum_{i,j} |\langle \boldsymbol{l}_i, \boldsymbol{r}_j\rangle| \leq \sqrt{mn}||\boldsymbol{B}||.$$

Notice that in both previous theorems the statements are true for all possible linear arrangements $\boldsymbol{l}_i$ and $\boldsymbol{r}_j$ (regardless of their dimensionalities). Also, notice that in Theorem 4.15 the matrix $\tilde{\boldsymbol{M}}$ can be picked independently of the linear arrangement and thus an adequate choice of $\tilde{\boldsymbol{M}}$ can make the bound much tighter.

More results concerning problems related to the ones above can be found in the works of Linial and Shraibman [2008], Forster et al. [2001] and Linial et al. [2007].

4.7 Applications in Data Science

Using the result from Theorem 4.3 sign rank becomes interesting in different applications from data mining and machine learning. It implies that the sign rank gives a lower bound for the number of dimensions we need to keep in dimensionality reduction methods. It also gives a lower bound on the number of attributes when one wants to perform data collection.

Let us start by looking at dimensionality reduction methods in a multi-class classification problem: Consider a binary linear classification problem with m points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \mathbb{R}^d$ and n classes $C_1, \ldots, C_n \subseteq \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ (where we do allow that a point belongs to multiple classes). Since $\mathbb{R}^d$ is a very high-dimensional space, we want to project the points $\boldsymbol{x}_i$ into a lower-dimensional space $\mathbb{R}^k$ in order to speed up our classification algorithms, which usually have running times growing in the number of features passed per data point. Assume that we perform the dimensionality reduction using a mapping $f : \mathbb{R}^d \to \mathbb{R}^k$. Now what is the smallest k, such that there exists a dimensionality reduction function $f : \mathbb{R}^d \to \mathbb{R}^k$, which satisfies that a linear classifier can classify the points $f(\boldsymbol{x}_i)$ with perfect accuracy according to the given classes?

The answer to this question will be the sign rank. We can define a binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$ by setting $\boldsymbol{B}_{ij} = 1$ iff $\boldsymbol{x}_i \in C_j$. Theorem 4.3 characterises the sign rank of $\boldsymbol{B}$ as the least number k, such that we can strictly linearly separate the classes C_i in the Euclidean space $\mathbb{R}^k$. Thus, since the points in Theorem 4.3 can be picked arbitrarily, the dimensionality reduction function f cannot be better than picking some points that can achieve the sign rank. Therefore, sign-rank($\boldsymbol{B}$) is equal to the least number of dimensions we need to keep when we want to separate the given classes with linear classifiers and perfect accuracy (no matter which kind of dimensionality reduction technique we choose).

In practise, feasible dimensionality reduction techniques will not be able to find a function f mapping into the space given by the sign rank (since these techniques would have to solve the sign rank problem, which is NP-hard as we had seen in Section 4.3). Thus, let us agree that the sign rank of the matrix $\boldsymbol{B}$ should rather be seen as a lower bound

on the number of dimensions to keep when we want to solve the above problem. Also, in practice we will often not be interested in classification with perfect accuracy (as this would often mean overfitting). Nonetheless, we can still use the sign rank or an approximation of it in order to gain some intuition into how difficult the data is to classify: Clearly, for data with high sign rank it would not make sense to perform the classification based on only very few features, while for data with small sign rank a small number of features would suffice.

It turns out that the sign rank is also interesting when we want to perform data collection. Assume that we are given some items of which we have not yet fetched any attributes, but we do know into which classes they belong. Then we can define a binary matrix $\boldsymbol{B}$ similar as above. Again using Theorem 4.3 we can see that the sign rank gives us as lower bound on the number of attributes we have to fetch on each item when we want to perform linear classification later on: As the sign rank gives us the smallest dimensionality, in which we can classify points of the given classes perfectly, real data should need more features.

To this better understand the problem let us consider the following toy example: Assume you are working for an American video game developer and you are developing a football simulation video game named after a large football association. Now you want to find out how many different skills you have to assign to each football player, in order to correctly classify him to the different positions he may play. Then the $\boldsymbol{x}_i$ of Theorem 4.3 would correspond to the players, e.g. Kevin Großkreutz, Mats Hummels and Marco Reus, and the classes would correspond to the positions the players can play, e.g. Central Back, Left Winger and Right Back. Now we know that Kevin Großkreutz can play all of these positions, Hummels can only play as a Central Back and Reus would only be useful as a Left Winger. Now the sign rank of this matrix tells you how many attributes you must at least gather per player.

Notice that here again the sign rank should only be treated as a lower bound on the number of attributes to fetch, since real data will contain noise and since the data points given by the real data will in general be more complex to linearly separate than the points that achieve the sign rank. Also, the results from the sign rank do not tell you which attributes you would need to fetch.

Chapter 5

Rounding Rank

This chapter will generalise sign rank as introduced in Chapter 5 by relaxing the notion of taking the sign to using rounding instead. This new rank will be called the *rounding rank*. After proving that this generalisation can only decrease the rank by at most one, we will further compare sign rank and rounding rank in Section 5.3, which will show that they are very similar. We will also derive a similar result for the *non-negative rounding rank*, which only allows non-negative factorisations. The chapter will be concluded by some examples of matrices for which we know their exact rounding ranks or which are interesting since they have very high rounding rank.

5.1 Definition and Characterisation

This section will formally introduce the rounding rank and argue how it generalises the sign rank. We will also see how the rounding rank relates to affine hyperplanes and affine hyperplane arrangements.

To be able to give the definition of rounding rank, let us start by introduction a rounding function. Let $\tau \in \mathbb{R}$ be arbitrary. Then we introduce the function

$$\operatorname{round}_\tau : \mathbb{R} \to \{0,1\}, \qquad z \mapsto \begin{cases} 1, & \text{if } z \geq \tau, \\ 0, & \text{if } z < \tau. \end{cases}$$

We will call τ the *rounding threshold*; we round to 1, if the value is larger than τ, and to 0, if the value is smaller than τ.

Now we overload the function $\operatorname{round}_\tau(\cdot)$ for matrices: Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be a real-valued matrix. Then we will write $\operatorname{round}_\tau(\boldsymbol{A})$ to denote the $m \times n$ binary matrix, which in each

entry agrees with the corresponding rounded entry of $\boldsymbol{A}$, i.e. for all $i = 1, \dots, m$ and $j = 1, \dots, n$, we have $(\text{round}_\tau(\boldsymbol{A}))_{ij} = \text{round}_\tau(\boldsymbol{A}_{ij})$.

Now we can define the *rounding rank of a binary matrix* as the smallest rank of any real-valued matrix, which in each entry rounds to the binary matrix's corresponding entry.

Definition 5.1. *Given a binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$ and a rounding threshold $\tau \in \mathbb{R}$, the* rounding rank of $\boldsymbol{B}$ with respect to τ *is given by the smallest $k \in \mathbb{N}$, such that there exists a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with* $\text{rank}(\boldsymbol{A}) = k$ *and* $\text{round}_\tau(\boldsymbol{A}) = \boldsymbol{B}$*. We denote the rounding rank of $\boldsymbol{B}$ by* $\text{rrank}_\tau(\boldsymbol{B})$.

In the latter part of the thesis will often drop the parameter τ and just write $\text{rrank}(\boldsymbol{B})$, in this case we assume the rounding threshold to be $\tau = \frac{1}{2}$.

Notice that the rounding rank generalises the sign rank: Let $\boldsymbol{A}$ be a real-valued matrix without zero-entries and let $\boldsymbol{B}^{\pm} = \text{sign}(\boldsymbol{A})$. Now denote the binary version of $\boldsymbol{B}^{\pm}$ by $\boldsymbol{B}$. Then it is easy to see that we have $\boldsymbol{B} = \text{round}_0(\boldsymbol{A})$. Due to this observation we obtain that the sign rank and the rounding rank with rounding threshold $\tau = 0$ coincide, i.e. we have $\text{rrank}_0(\boldsymbol{B}) = \text{sign-rank}(\boldsymbol{B})$.

In Section 4.2 we had seen that sign rank and hyperplane arrangements are equivalent. Now we will shortly argue how rounding rank and affine hyperplane arrangements relate.

An *affine hyperplane arrangement in* $\mathbb{R}^d$ is a set $\mathcal{A} = \{H_1, \dots, H_n\}$ consisting of n affine hyperplanes of the form $H_i = \{\boldsymbol{x} \in \mathbb{R}^d : \langle \boldsymbol{c}_i, \boldsymbol{x} \rangle = b_i\}$, where the $\boldsymbol{c}_i$ are called the *normal vectors* and the scalars b_i are called the *offsets from the origin*. Notice that such an affine hyperplane will only contain the origin if $b_i = 0$.

The sign vectors of the affine hyperplane arrangement can be defined as in Section 4.2 using the sets H_i^+ and H_i^- after replacing zero with the offset from the origin. For us it will be more useful to express them using scalar products. To do this, let $\boldsymbol{x} \in \mathbb{R}^d$ be a point. Then the *sign vector $t \in \{-, 0, +\}^n$ of $\boldsymbol{x}$* is given by the vector

$$t = (\text{sign}(\langle \boldsymbol{x}, \boldsymbol{c}_1 \rangle - b_1), \dots, \text{sign}(\langle \boldsymbol{x}, \boldsymbol{c}_n \rangle - b_n)).$$

The *set of sign vectors*, $\mathcal{V}(\mathcal{A})$, is defined as before. In the latter of this chapter we will often consider the sign vectors to be binary. We can achieve this by mapping 0 and + to 1 and − to 0.

Remembering the characterisation of sign rank from Theorem 4.3, we can now give a similar result for rounding rank.

Theorem 5.2. *Let $\boldsymbol{B} \in \{0,1\}^{m\times n}$ be a binary matrix and let $d \in \mathbb{N}$. Then the following statements are equivalent:*

1. *$\boldsymbol{B}$ has rounding rank at most d for some non-zero rounding threshold, i.e. we have $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$ for some $\tau \neq 0$.*
2. *There exist points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, such that for all $j = 1, \ldots, m$, the classes $C_j = \{\boldsymbol{x}_i : \boldsymbol{B}_{ij} = 1\}$ and $\bar{C}_j = \{\boldsymbol{x}_i : \boldsymbol{B}_{ij} = 0\}$ are strictly linearly separable with affine hyperplanes.*
3. *There exists an arrangement of affine hyperplanes, $\mathcal{A} = \{H_1, \ldots, H_n\}$, in $\mathbb{R}^d$ with $\boldsymbol{B} \subseteq \mathcal{V}(\mathcal{A})$, i.e. the rows of $\boldsymbol{B}$ are a subset of the set sign vectors of $\mathcal{A}$.*

We omit the proof since it would be very similar to the proof of Theorem 4.3. The major difference is that at the right places we would need to apply two results that we will encounter later in this chapter. These results are Corollary 5.5 and Lemma A.2 and their usages are similar to how they will be used them in the proof of Theorem 5.8.

Having seen Theorem 5.2 and how rounding rank and affine hyperplane arrangements correspond to one another, we observe that the offset from the origin b of an affine hyperplane with normal vector $\boldsymbol{c}$ and the rounding threshold of the rounding rank basically describe the same quantity: An entry of the sign vector of a point $\boldsymbol{x}$ is given by an equation of the form $\operatorname{sign}(\langle \boldsymbol{x}, \boldsymbol{c}\rangle - b)$. For the rounding we compute $\operatorname{round}_b(\langle \boldsymbol{x}, \boldsymbol{c}\rangle)$, which is the same as $\operatorname{round}_0(\langle \boldsymbol{x}, \boldsymbol{c}\rangle - b)$. Thus, these two notions are the same.

5.2 Changing the Rounding Threshold

It is a natural question to ask how the rounding rank behaves for different rounding thresholds. In this section we will see that changing the rounding threshold can alter the rounding rank by at most one and under which conditions this happens. A corollary from this will show that sign rank and rounding rank will differ by at most one.

We start by giving a Lemma for exchanging the rounding threshold at the cost of increasing the rank by one.

Lemma 5.3. *Let $\boldsymbol{B} \in \{0,1\}^{m\times n}$ be some binary matrix with $\operatorname{rrank}_\tau(\boldsymbol{B}) = r$ for some $\tau \in \mathbb{R}$. Then for all $\tau' \in \mathbb{R}$ we have $\operatorname{rrank}_{\tau'}(\boldsymbol{B}) \leq r + 1$.*

Proof. Let $\tau' \in \mathbb{R}$ be arbitary. Let $\boldsymbol{L}\boldsymbol{R}^T$ be a rounding rank r decomposition with rounding threshold τ. Set $c = \tau' - \tau$ and observe that

$$\begin{aligned}\boldsymbol{B}_{ij} &= \operatorname{round}_\tau([\boldsymbol{L}\boldsymbol{R}^T]_{ij}) \\ &= \operatorname{round}_{\tau+c}([\boldsymbol{L}\boldsymbol{R}^T]_{ij} + c) \\ &= \operatorname{round}_{\tau'}([\boldsymbol{L}\boldsymbol{R}^T]_{ij} + c).\end{aligned}$$

Thus, for $\boldsymbol{L}' = \begin{pmatrix}\boldsymbol{L} & c\mathbf{1}\end{pmatrix} \in \mathbb{R}^{m\times(r+1)}$ and $\boldsymbol{R}' = \begin{pmatrix}\boldsymbol{R} & \mathbf{1}\end{pmatrix} \in \mathbb{R}^{n\times(r+1)}$, we obtain for the new matrices $\operatorname{round}_{\tau'}(\boldsymbol{L}'\boldsymbol{R}'^T) = \boldsymbol{B}$ and thus $\operatorname{rrank}_{\tau'}(\boldsymbol{B}) \leq r+1$. □

Now we will continue to see under which circumstances the result from Lemma 5.3 can be improved. We will see the exact conditions under which we can avoid gaining rank and that in general this cannot be avoided.

Before we can give these main findings of this section, we will need some results from convex geometry. We start by looking at the famous Hyperplane Separation Theorem, which states a condition under which two sets can be strictly linearly separated by an affine hyperplane.

Theorem 5.4 (Hyperplane Separation Theorem, [Boyd and Vandenberghe, 2004, page 46])**.** *Let A and B be two disjoint nonempty closed convex sets in $\mathbb{R}^d$, one of which is compact. Then there exists a nonzero vector $\boldsymbol{v} \in \mathbb{R}^d$ and real numbers $c_1 < c_2$, such that $\langle \boldsymbol{x}, \boldsymbol{v}\rangle > c_2$ and $\langle \boldsymbol{y}, \boldsymbol{v}\rangle < c_1$ for all $\boldsymbol{x} \in A$ and $\boldsymbol{y} \in B$.*

Notice that the theorem implies that if we look at an affine hyperplane H with normal vector $\boldsymbol{v}$ and offset from the origin $b \in (c_1, c_2)$, then H will strictly separate the sets A and B. As we had seen in Section 5.1, this offset from the origin is basically the same quantity as the rounding threshold τ. Thus, it is somewhat unhandy that the numbers c_1 and c_2 cannot be picked beforehand, because the rounding threshold τ may not be from the open interval (c_1, c_2). The following corollary shows that this issue can be fixed for non-zero rounding thresholds.

Corollary 5.5. *Let A and B be two disjoint nonempty convex sets in $\mathbb{R}^d$, one of which is compact. Then for all $c \in \mathbb{R} \setminus \{0\}$ there exists a nonzero vector $\boldsymbol{v} \in \mathbb{R}^d$, such that $\langle \boldsymbol{x}, \boldsymbol{v}\rangle > c$ and $\langle \boldsymbol{y}, \boldsymbol{v}\rangle < c$ for all $\boldsymbol{x} \in A$ and $\boldsymbol{y} \in B$.*

Proof. Let $c \in \mathbb{R} \setminus \{0\}$ be arbitrary. We apply Theorem 5.4 to A and B to obtain a vector $\boldsymbol{v}'$ and numbers $c_1 < c_2$ with $\langle \boldsymbol{x}, \boldsymbol{v}'\rangle > c_2$ and $\langle \boldsymbol{y}, \boldsymbol{v}'\rangle < c_1$ for all $\boldsymbol{x} \in A$ and all $\boldsymbol{y} \in B$. Now we consider three cases.

Case 1: $c_1 \neq 0$ and $c_2 \neq 0$ and $\text{sign}(c_1) = \text{sign}(c_2)$. We set $\alpha = c/c_2$ and $\boldsymbol{v} = \alpha \boldsymbol{v}'$. Then we get for $\boldsymbol{x} \in A$:

$$\langle \boldsymbol{x}, \boldsymbol{v} \rangle = \alpha \langle \boldsymbol{x}, \boldsymbol{v}' \rangle > \alpha c_2 = c,$$

as well as for $\boldsymbol{y} \in B$:

$$\langle \boldsymbol{y}, \boldsymbol{v} \rangle = \alpha \langle \boldsymbol{y}, \boldsymbol{v}' \rangle < \alpha c_1 = \frac{c_1}{c_2} c < c,$$

where in the last inequality we used that $0 < c_1/c_2 < 1$.

Case 2: $c_1 \neq 0$ and $c_2 \neq 0$ and $\text{sign}(c_1) \neq \text{sign}(c_2)$. Since the signs of c_1 and c_2 disagree, we have $c_1 < 0 < c_2$. Thus, we can pick $c_1' \in (0, c_2)$ arbitrarily and still maintain all properties guaranteed by the Hyperplane Separation Theorem for $\boldsymbol{v}$, c_1' and c_2. Now we are in case 1.

Case 3: $c_1 = 0$ or $c_2 = 0$. We can pick numbers $d_1, d_2 \in (c_1, c_2)$ with $d_1 < d_2$. Observe that both d_1 and d_2 are non-zero. Then we have $\langle \boldsymbol{x}, \boldsymbol{v}' \rangle > c_2 > d_2$ and $\langle \boldsymbol{y}, \boldsymbol{v}' \rangle < c_1 < d_1$ for all $\boldsymbol{x} \in A$ and all $\boldsymbol{y} \in B$. Now we can use case 1 for $\boldsymbol{v}'$, d_1 and d_2. □

Having seen Corollary 5.5, we are ready to prove under which condition we can avoid increasing the rank when changing the rounding threshold. This condition is given in the next definition.

Definition 5.6. *We call a binary matrix* $\boldsymbol{B} \in \{0,1\}^{m \times n}$ thresholdable*, if it contains no all-zero and no all-one columns.*

To see why the notion of thresholdable matrices is useful, let us look at an example. The example will show that for a non-thresholdable matrix we cannot avoid increasing the rank when changing the rounding threshold.

Example 5.7. *We present a non-thresholdable matrix* $\boldsymbol{B}$*, that gains rank after changing the rounding threshold. We set*

$$\boldsymbol{B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \qquad \boldsymbol{L} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \qquad \boldsymbol{R} = \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix}.$$

Notice that $\boldsymbol{B}$ *is not thresholdable, since its third column contains only zeros. Further observe that* $\text{rrank}_{\frac{1}{2}}(\boldsymbol{B}) = 1$*, since* $\boldsymbol{B} = \text{round}(\boldsymbol{L}\boldsymbol{R}^T)$.

Now we will see that for a negative rounding threshold $\tau < 0$ *we have* $\text{rrank}_\tau(\boldsymbol{B}) > 1$*. Assume (for contradiction) that there exists a rounding rank one decomposition* $\boldsymbol{L}\boldsymbol{R}^T$

with

$$\boldsymbol{L} = \begin{pmatrix} \boldsymbol{l}_1 \\ \boldsymbol{l}_2 \end{pmatrix}, \qquad \boldsymbol{R} = \begin{pmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \end{pmatrix}.$$

Since $\boldsymbol{L}\boldsymbol{R}^T$ *is a rounding rank decomposition, it must satisfy the following inequalities:*

$$\begin{aligned} \boldsymbol{l}_1\boldsymbol{r}_1 \geq \tau, & \qquad & \boldsymbol{l}_2\boldsymbol{r}_1 < \tau < 0, \\ \boldsymbol{l}_1\boldsymbol{r}_2 < \tau < 0, & & \boldsymbol{l}_2\boldsymbol{r}_2 \geq \tau, \\ \boldsymbol{l}_1\boldsymbol{r}_3 < \tau < 0, & & \boldsymbol{l}_2\boldsymbol{r}_3 < \tau < 0. \end{aligned}$$

Let us first assume that $\boldsymbol{r}_1 \neq 0 \neq \boldsymbol{r}_2$*. Then without loss of generality we can assume that we have* $\operatorname{sign}(\boldsymbol{l}_1) = \operatorname{sign}(\boldsymbol{l}_2) = -$ *and* $\operatorname{sign}(\boldsymbol{r}_1) = \operatorname{sign}(\boldsymbol{r}_2) = \operatorname{sign}(\boldsymbol{r}_3) = +$*. Rewriting the above inequalities we get the following inequalities:*

$$\frac{\tau}{\boldsymbol{r}_2} > \boldsymbol{l}_1 \geq \frac{\tau}{\boldsymbol{r}_1}, \qquad \frac{\tau}{\boldsymbol{r}_1} > \boldsymbol{l}_2 \geq \frac{\tau}{\boldsymbol{r}_2}.$$

Now consider the case that $\boldsymbol{r}_1 = 0$*. Then we get* $\boldsymbol{l}_2\boldsymbol{r}_1 = 0 \not< \tau < 0$*. Also, for* $\boldsymbol{r}_2 = 0$*, we have* $\boldsymbol{l}_1\boldsymbol{r}_2 = 0 \not< \tau < 0$*. Hence, we must have* $\boldsymbol{r}_1 \neq 0 \neq \boldsymbol{r}_2$*.*

Combining the three cases, we obtain a contradiction, since there exists no satisfying assignment for $\boldsymbol{r}_1$ *and* $\boldsymbol{r}_2$*. Therefore, we have* $\operatorname{rrank}_\tau(\boldsymbol{B}) > 1$*. From Lemma 5.3 we obtain* $\operatorname{rrank}_\tau(\boldsymbol{B}) = 2$*.*

This example showed that we may gain rank, when the given binary matrix is not thresholdable. So, what happens for thresholdable matrices? The following Theorem shows that if the matrices are thresholdable, then we can change to arbitrary non-zero rounding thresholds.

Theorem 5.8. *Let* $\boldsymbol{B} \in \{0,1\}^{m \times n}$ *be a thresholdable binary matrix and let* $\boldsymbol{L}\boldsymbol{R}^T$ *be a rounding rank* k *decomposition of* $\boldsymbol{B}$ *with rounding threshold* τ*. Then for all* $\tau' \neq 0$ *there exists a rounding rank* k *decomposition* $\boldsymbol{L}\boldsymbol{R}'^T$ *with rounding threshold* τ'*.*

Proof. Let $\tau' \in \mathbb{R} \setminus \{0\}$ be arbitrary. We will consider the rows of $\boldsymbol{L}$ as points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ in $\mathbb{R}^k$ and construct a new $n \times k$ matrix $\boldsymbol{R}'$ consisting of normal vectors of hyperplanes in $\mathbb{R}^k$ in its rows, that separate the points with the correct rounding threshold.

To construct the j'th row of $\boldsymbol{R}'$, let $C_j = \{\boldsymbol{l}_i : \boldsymbol{B}_{ij} = 1\}$ and let its complement be given by $\bar{C}_j = \{\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m\} \setminus C = \{\boldsymbol{l}_i : \boldsymbol{B}_{ij} = 0\}$. Now we observe that by Lemma A.2 the

convex hulls of C_j and $\bar{C}_j$ are separated by the affine hyperplane with the j'th row of $\boldsymbol{R}$ as its normal vector and offset from the origin τ.

Thus, we can apply Corollary 5.5 to obtain a vector $\boldsymbol{r}'_j$, such that $\langle \boldsymbol{r}'_j, \boldsymbol{c} \rangle > \tau'$ for all $\boldsymbol{c} \in C_j$ and $\langle \boldsymbol{r}'_j, \boldsymbol{c} \rangle < \tau'$ for all $\boldsymbol{c}' \in \bar{C}$. Now we set $\boldsymbol{r}'_j$ to be the j'th row of $\boldsymbol{R}'$.

Repeating this procedure for all n rows of $\boldsymbol{R}'$ proves the claim, since then we have $\text{round}_{\tau'}(\boldsymbol{L}\boldsymbol{R}'^T) = \boldsymbol{B}$. □

From the theorem we can immediately derive a corollary, which proves that for thresholdable matrices the rounding rank does not change for non-zero rounding thresholds.

Corollary 5.9. *Let $\boldsymbol{B}$ be a thresholdable binary matrix. Then non-zero rounding thresholds do not influence the rounding rank of a matrix, i.e. for all $\tau, \tau' \in \mathbb{R} \setminus \{0\}$, we have* $\text{rrank}_\tau(\boldsymbol{B}) = \text{rrank}_{\tau'}(\boldsymbol{B})$. *We also have* $\text{rrank}_0(\boldsymbol{B}) \geq \text{rrank}_\tau(\boldsymbol{B})$ *for all τ.*

Proof. For the first claim let $\tau \neq 0 \neq \tau'$. Without loss of generality assume that $\text{rrank}_\tau(\boldsymbol{B}) = k$ and that $\text{rrank}_{\tau'}(\boldsymbol{B}) = l > k$. Then we can apply Theorem 5.8 to obtain a decomposition with threshold τ' from the decomposition with threshold τ. Thus $\text{rrank}_{\tau'}(\boldsymbol{B}) \leq k$. A contradiction.

The second claim of the corollary works by directly applying Theorem 5.8. □

Observe that Theorem 5.8 allows us to take a rounding rank factorisation with rounding threshold zero and to transform it into one with non-zero rounding threshold. But it did not offer us a solution if the new rounding rank τ' was supposed to be zero. The following example shows that in general this is not possible.

Example 5.10. *In this example we construct a binary matrix for which its sign rank is bigger than its rounding rank by using the theory of hyperplane arrangements.*

Let $\mathcal{A}$ and $\mathcal{A}'$ be hyperplane arrangements in $\mathbb{R}^d$ with $n \leq d$ hyperplanes, where $\mathcal{A}$ contains affine hyperplanes and $\mathcal{A}'$ has only hyperplanes through the origin. Also, assume that from all hyperplanes arrangements with these properties, $\mathcal{A}$ and $\mathcal{A}'$ are the ones that maximise the number of regions into which they intersect $\mathbb{R}^d$ (and thus they also maximise the number of their sign vectors).

Edelsbrunner [1987, see section 1.2] proves that the affine hyperplane arrangement $\mathcal{A}$ dissects the space into $\sum_{i=0}^{d} \binom{n}{i} = O(n^d)$ regions, while $\mathcal{A}'$ dissects the space only into $2^n \leq 2^d$ regions[1]*. Conclusively, $\mathcal{A}$ dissects the space into strictly more regions than $\mathcal{A}'$ and therefore $\mathcal{A}$ will support strictly more sign vectors than $\mathcal{A}'$.*

[1] It also possible (and maybe easier, but certainly less complicated) to draw an example in $\mathbb{R}^2$ with three affine hyperplanes, which intersect the space into seven regions. Then observe that three hyperplanes through the origin, can intersect the plane into at most six regions.

Writing all sign vectors of $\mathcal{A}$ into a sign matrix $\boldsymbol{B}$, we have found a matrix with $\text{sign-rank}(\boldsymbol{B}) > \text{rrank}(\boldsymbol{B})$.

To conclude the section let us summarise that in most cases the rounding threshold does not play a role. The cases when it matters are rounding rank zero (i.e. the sign rank) and non-thresholdable matrices, but even then it only changes the rank by at most one. These results are tight (i.e. they cannot be improved) as our previous examples proved.

5.3 Comparison to Sign Rank

Sign rank and rounding rank are not much different. The results from the previous section showed us that for a given binary matrix, the sign rank and the rounding rank differ by at most one. The following theorem formalises and proves this claim.

Theorem 5.11. *Rounding rank and sign rank differ by at most 1, i.e. for all binary matrices $\boldsymbol{B}$ we have*

$$\text{rrank}(\boldsymbol{B}) \leq \text{sign-rank}(\boldsymbol{B}) \leq \text{rrank}(\boldsymbol{B}) + 1.$$

Proof. The first inequality is just the second claim of Corollary 5.9, since by definition we have $\text{rrank}_0(\boldsymbol{B}) = \text{sign-rank}(\boldsymbol{B})$. The second inequality follows from Lemma 5.3. □

This theorem immediately implies that all upper and lower bounds that we saw in Section 4.4 for sign rank also hold for rounding rank (where one may have to add or subtract 1 accordingly).

In terms of computational complexity the results of Kang and Müller [2012] imply that for all $k \geq 2$ it is NP-hard to decide whether a binary matrix has rounding rank at most k or not (with respect to rounding threshold $\tau = 1$).

As we already mentioned in the first section of this chapter, rounding rank is also closely related to hyperplane arrangements. The only difference is that when sign rank considers hyperplanes containing the origin, rounding rank considers affine hyperplanes with the offset from the origin given by the corresponding rounding threshold.

5.4 Non-negative Rounding Rank

It is also interesting to analyse how the non-negative rounding rank of a binary matrix behaves compared to the rounding rank, i.e. what happens if we only allow non-negative

factor matrices for the rounding rank decompositions. In this section we will prove that the rounding rank and the non-negative rounding rank differ by at most two. We will obtain this result by giving a simple, but somewhat technical construction.

Let us first formally define the non-negative rounding rank. For this purpose let us denote the set of all non-negative real numbers by $\mathbb{R}_+$, i.e. $\mathbb{R}_+ = \{x \in \mathbb{R} : x \geq 0\}$.

Definition 5.12. *The* non-negative rounding rank *of a binary matrix* $\boldsymbol{B} \in \{0,1\}^{m \times n}$*, denoted by* $\operatorname{rrank}_+(\boldsymbol{B})$*, is the smallest* $k \in \mathbb{N}$*, such that there exist matrices* $\boldsymbol{L} \in \mathbb{R}_+^{m \times k}$ *and* $\boldsymbol{R} \in \mathbb{R}_+^{n \times k}$ *with* $\boldsymbol{B} = \operatorname{round}(\boldsymbol{L}\boldsymbol{R}^T)$*.*

Now we are able to state and prove the main result of this section. The proof follows ideas of Paturi and Simon [1984] and adds some details to achieve the non-negativity.

Theorem 5.13. *Let* $\boldsymbol{B} \in \{0,1\}^{m \times n}$ *be a binary matrix. Then we have*

$$\operatorname{rrank}(\boldsymbol{B}) \leq \operatorname{rrank}_+(\boldsymbol{B}) \leq \operatorname{rrank}(\boldsymbol{B}) + 2.$$

Proof. The first inequality is trivial as the standard rounding rank is more general than the non-negative rounding rank.

The trickier part is the second inequality. The idea of the proof is to take points and hyperplanes achieving the rounding rank of the matrix and to project them into a higher dimensional space, where they are non-negative. This projection happens via an explicit construction, that gets somewhat technical.

Let $k = \operatorname{rrank}(\boldsymbol{B})$. Then by definition there exist matrices $\boldsymbol{L} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{R} \in \mathbb{R}^{n \times k}$ with $\boldsymbol{B} = \operatorname{round}(\boldsymbol{L}\boldsymbol{R}^T)$ for rounding threshold $\frac{1}{2}$. As before we will interpret the rows $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ of $\boldsymbol{L}$ as points in $\mathbb{R}^k$ and the rows $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n$ of $\boldsymbol{R}$ as normal vectors of affine hyperplanes in $\mathbb{R}^k$.

For each $\boldsymbol{r}_j = (\boldsymbol{r}_{j1}, \ldots, \boldsymbol{r}_{jk})$, we set $\boldsymbol{r}'_j = \left(\boldsymbol{r}_{j1}, \ldots, \boldsymbol{r}_{jk}, -\frac{1}{2}, \frac{1}{2} - \sum_{m=1}^{k} \boldsymbol{r}_{jm}\right) \in \mathbb{R}^{k+2}$ and observe that these vectors define hyperplanes in $\mathbb{R}^{k+2}$ containing the origin, i.e. we have $\mathbf{0} \in \left\{\boldsymbol{x} \in \mathbb{R}^{k+2} : \langle \boldsymbol{x}, \boldsymbol{r}'_j \rangle = 0\right\}$. We set $d_j = \max\{|\boldsymbol{r}'_{jm}| : m = 1, \ldots, k+2\}$ and define $\boldsymbol{r}''_j = \frac{1}{2d_j}\boldsymbol{r}'_j$. Observe that for all $m = 1, \ldots, k+2$, we have $-\frac{1}{2} \leq \boldsymbol{r}''_{jm} \leq \frac{1}{2}$.

For each $\boldsymbol{l}_i = (\boldsymbol{l}_{i1}, \ldots, \boldsymbol{l}_{ik})$, we set $c_i = \max\{|\boldsymbol{l}_{i1}|, \ldots, |\boldsymbol{l}_{ik}|, 1\}$ and we further define $\boldsymbol{l}'_i = (c_i + \boldsymbol{l}_{i1}, \ldots, c_i + \boldsymbol{l}_{ik}, c_i + 1, c_i) \in \mathbb{R}^{k+2}$ and observe that $\boldsymbol{l}'_i$ is non-zero and non-negative. By $\boldsymbol{l}''_i$ we denote $\boldsymbol{l}'_i$ after normalizing with the L^1-norm, i.e. $\boldsymbol{l}''_i = \boldsymbol{l}'_i / ||\boldsymbol{l}'_i||_1$, where $||\boldsymbol{l}'_i||_1 = \sum_{m=1}^{k+2} |\boldsymbol{l}_{im}|$.

Now we do a short intermediate computation that shows that the $\boldsymbol{l}_i''$ and $\boldsymbol{r}_j''$ indeed still round to matrix $\boldsymbol{B}$ with rounding threshold 0:

$$\begin{aligned}
\langle \boldsymbol{r}_j'', \boldsymbol{l}_i'' \rangle &= \frac{1}{||\boldsymbol{l}_i'||_1} \langle \boldsymbol{r}_j'', \boldsymbol{l}_i' \rangle \\
&= \frac{1}{2d_j||\boldsymbol{l}_i'||_1} \langle \boldsymbol{r}_j', \boldsymbol{l}_i' \rangle \\
&= \frac{1}{2d_j||\boldsymbol{l}_i'||_1} \left(\sum_{m=1}^{k} \boldsymbol{r}_{jm}(c_i + \boldsymbol{l}_{im}) - \frac{1}{2}(c_i + 1) + \left(\frac{1}{2} - \sum_{m=1}^{k} \boldsymbol{r}_{jm} \right) c_i \right) \\
&= \frac{1}{2d_j||\boldsymbol{l}_i'||_1} \left(\sum_{m=1}^{k} \boldsymbol{r}_{jm}\boldsymbol{l}_{im} - \frac{1}{2} \right) \\
&= \frac{1}{2d_j||\boldsymbol{l}_i'||_1} \left(\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle - \frac{1}{2} \right) \qquad (5.1) \\
&= \begin{cases} \geq 0, & \text{if } \langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle \geq \frac{1}{2}, \\ < 0, & \text{otherwise.} \end{cases}
\end{aligned}$$

We move on to define $\boldsymbol{r}_j''' \in \mathbb{R}^{k+2}$ by setting $\boldsymbol{r}_{jl}''' = \frac{1}{2} + \boldsymbol{r}_{jl}''$ for all $l = 1, \ldots, k+2$. Observe that each component of $\boldsymbol{r}_j'''$ is non-negative. We perform another intermediate computation, that we will need later:

$$\begin{aligned}
\langle \left(\frac{1}{2}, \ldots, \frac{1}{2} \right), \boldsymbol{l}_i'' \rangle &= \frac{1}{2} \sum_{m=1}^{k+2} \boldsymbol{l}_{im}'' \\
&= \frac{1}{2||\boldsymbol{l}_i'||_1} \sum_{m=1}^{k+2} \boldsymbol{l}_{im}' \\
&= \frac{1}{2||\boldsymbol{l}_i'||_1} \left(\sum_{m=1}^{k} (c_i + \boldsymbol{l}_{im}) + 2c_i + 1 \right) \\
&= \frac{1}{2||\boldsymbol{l}_i'||_1} \left((k+2)c_i + 1 + \sum_{m=1}^{k} \boldsymbol{l}_{im} \right). \qquad (5.2)
\end{aligned}$$

Now we observe that the $\boldsymbol{r}_j'''$ and $\boldsymbol{l}_i''$ give a non-negative rounding rank decomposition of $\boldsymbol{B}$ for different rounding thresholds, where we use equations 5.1 and 5.2 in the second step:

$$\begin{aligned}
\langle \boldsymbol{r}_j''', \boldsymbol{l}_i'' \rangle &= \langle \boldsymbol{r}_j'' + \left(\frac{1}{2}, \ldots, \frac{1}{2} \right), \boldsymbol{l}_i'' \rangle \\
&= \frac{\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle - \frac{1}{2}}{2d_j||\boldsymbol{l}_i'||_1} + \frac{(k+2)c_i + 1 + \sum_{m=1}^{k} \boldsymbol{l}_{im}}{2||\boldsymbol{l}_i'||_1}. \qquad (5.3)
\end{aligned}$$

Notice that the first summand of equation 5.3 is non-negative iff $\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle \geq \frac{1}{2}$. Thus, if we

use the second summand as rounding threshold, then we would round correctly. The issue is that this rounding threshold depends on $\boldsymbol{l}_i$.

To solve this problem and to get everything to rounding threshold $\frac{1}{2}$, we rescale the $\boldsymbol{l}_i''$. We denote the second summand of equation 5.3 by α and observe that $\alpha \geq 0$ by choice of c_i. Now we set $\boldsymbol{l}_i''' = \frac{1}{2\alpha}\boldsymbol{l}_i''$ and obtain:

$$\begin{aligned}\langle \boldsymbol{r}_j''', \boldsymbol{l}_i''' \rangle &= \frac{1}{2\alpha}\langle \boldsymbol{r}_i''', \boldsymbol{l}_i'' \rangle \\ &= \frac{\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle - \frac{1}{2}}{4\alpha d_j ||\boldsymbol{l}_i'||_1} + \frac{1}{2}, \end{aligned} \tag{5.4}$$

where we used equation 5.3 in the last step. The inner product is non-negative by choice of $\boldsymbol{l}_i'''$ and $\boldsymbol{r}_j'''$ and the first summand of equation 5.4 is non-negative iff $\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle \geq \frac{1}{2}$. Thus, $\langle \boldsymbol{r}_j''', \boldsymbol{l}_i''' \rangle \geq \frac{1}{2}$ iff $\langle \boldsymbol{r}_j, \boldsymbol{l}_i \rangle \geq \frac{1}{2}$ iff $\boldsymbol{B}_{ij} = 1$. Therefore, the $\boldsymbol{r}_j'''$ and $\boldsymbol{l}_i'''$ give a non-negative rounding rank decomposition of $\boldsymbol{B}$ for rounding threshold $\frac{1}{2}$. □

After having seen that rounding rank and non-negative rounding rank are almost the same, we also want to give a characterisation of matrices with non-negative rounding rank one. To be able to give this characterisation, we need another definition.

Definition 5.14 (Nestedness, [Junttila, 2011, page 38])**.** *A binary matrix $\boldsymbol{B}$ is* directly nested *if for each one-entry $\boldsymbol{B}_{ij} = 1$, we have $\boldsymbol{B}_{i'j'} = 1$ for all $i' \in \{1, \ldots, i-1\}$ and $j' \in \{1, \ldots, j-1\}$.*

A binary matrix $\boldsymbol{B}$ is nested *if there exist permutation matrices*[2] *$\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, such that $\boldsymbol{P}_1\boldsymbol{B}\boldsymbol{P}_2$ is nested.*

It turns out that it is easy to check whether a matrix is nested. Junttila [2011, page 39] proves that a matrix is nested if and only if it does not contain any 2×2 submatrix of the form

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \text{or} \qquad \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Thus, the following Lemma, which characterises binary matrices with non-negative rounding rank one as nested matrices, will give us an easy certificate to check whether a matrix has non-negative rounding rank one.

Lemma 5.15. *Let $\boldsymbol{B} \in \{0,1\}^{m \times n}$. Then the following two statements are equivalent:*

1. *$\boldsymbol{B}$ is nested.*

[2] See Definition A.3.

2. $\boldsymbol{B}$ *has non-negative rounding rank one, i.e.* $\operatorname{rrank}_+(\boldsymbol{B}) = 1$.

Proof. $1 \Rightarrow 2$: Let $\boldsymbol{B}$ be nested and let $\boldsymbol{C}$ be its directly nested version after applying the permutation matrices $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$.

Set $\boldsymbol{p} = \boldsymbol{C}\mathbf{1}$, i.e. $\boldsymbol{p}$ is the vector containing the row sums of $\boldsymbol{C}$. Since $\boldsymbol{C}$ is directly nested, we have $\boldsymbol{p}_1 \geq \cdots \geq \boldsymbol{p}_n$. Now we set $\boldsymbol{l}_i = 2^{\boldsymbol{p}_i - 1}$ and $\boldsymbol{r}_j = 2^{-j}$. Then we get $\boldsymbol{l}_i\boldsymbol{r}_j = 2^{\boldsymbol{p}_i - j - 1}$ and thus we get that $\boldsymbol{l}_i\boldsymbol{r}_j \geq 1/2$ if and only if $j \leq \boldsymbol{p}_i$. Since $\boldsymbol{p}_i$ is the number of ones in the in the ith row of $\boldsymbol{C}$, we have $\boldsymbol{C}_{ij} = \operatorname{round}(\boldsymbol{l}_i\boldsymbol{r}_j)$.

By setting $\tilde{\boldsymbol{l}} = \boldsymbol{P}_1\boldsymbol{l}$ and $\tilde{\boldsymbol{r}}^T = \boldsymbol{r}^T\boldsymbol{P}_2$ we get $\tilde{\boldsymbol{l}}\tilde{\boldsymbol{r}}^T = \boldsymbol{P}_1\boldsymbol{l}\boldsymbol{r}^T\boldsymbol{P}_2 = \boldsymbol{P}_1\boldsymbol{C}\boldsymbol{P}_2 = \boldsymbol{B}$ and therefore $\operatorname{rrank}_+(\boldsymbol{B}) = 1$.

$2 \Rightarrow 1$: Let $\boldsymbol{l} \geq 0$ and $\boldsymbol{r} \geq 0$ be such that $\boldsymbol{B} = \operatorname{round}(\boldsymbol{l}\boldsymbol{r}^T)$. Then there exist permutation matrices $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, s.t. for $\tilde{\boldsymbol{l}} = \boldsymbol{P}_1\boldsymbol{l}$ we have $\tilde{\boldsymbol{l}}_1 \geq \cdots \geq \tilde{\boldsymbol{l}}_m$ and for $\tilde{\boldsymbol{r}}^T = \boldsymbol{r}^T\boldsymbol{P}_2$ we have $\tilde{\boldsymbol{r}}_1 \geq \cdots \geq \tilde{\boldsymbol{r}}_n$.

Set $\boldsymbol{C} = \operatorname{round}(\tilde{\boldsymbol{l}}\tilde{\boldsymbol{r}}^T)$. Now we observe that $\tilde{\boldsymbol{l}}_i\tilde{\boldsymbol{r}}_j \geq \tilde{\boldsymbol{l}}_{i+1}\boldsymbol{r}_j$ and therefore for each entry of $\boldsymbol{C}$ we also have $\boldsymbol{C}_{ij} = \operatorname{round}(\boldsymbol{l}_i\tilde{\boldsymbol{r}}_j) \geq \operatorname{round}(\tilde{\boldsymbol{l}}_{i+1}\boldsymbol{r}_j) = \boldsymbol{C}_{(i+1)j}$. Similarly, we obtain $\boldsymbol{C}_{ij} = \operatorname{round}(\boldsymbol{l}_i\tilde{\boldsymbol{r}}_j) \geq \operatorname{round}(\tilde{\boldsymbol{l}}_i\boldsymbol{r}_{j+1}) = \boldsymbol{C}_{i(j+1)}$. Therefore, $\boldsymbol{C}$ is directly nested.

We conclude that $\boldsymbol{B} = \boldsymbol{l}\boldsymbol{r}^T$ is nested, since $\boldsymbol{B} = \boldsymbol{l}\boldsymbol{r}^T = \boldsymbol{P}_1^{-1}\tilde{\boldsymbol{l}}\tilde{\boldsymbol{r}}^T\boldsymbol{P}_2^{-1} = \boldsymbol{P}_1^{-1}\boldsymbol{C}\boldsymbol{P}_2^{-1}$. □

5.5 Some Examples

In this section we will discuss some special matrices; some for which we know their exact rounding ranks and some which have a very large rounding rank. We will start by giving a characterisation of matrices with rounding rank one. After that we will continue to show that all identity matrices (no matter of their sizes) have rounding rank two. The reader is also referred to Lemma 5.15, which characterised nested matrices as matrices with non-negative rounding rank one. The section is concluded by the Hadamard matrices, which have very large sign rank.

We start by giving a characterisation of matrices with rounding rank 1.

Lemma 5.16. *Let* $\boldsymbol{B} \in \{0,1\}^{m\times n}$ *with* $\boldsymbol{B} \neq 0$. *Then the following two statements are equivalent:*

1. $\boldsymbol{B}$ *has rounding rank one, i.e.* $\operatorname{rrank}(\boldsymbol{B}) = 1$.

2. $\boldsymbol{B}$ *is nested or there exist permutation matrices* $\boldsymbol{P}_1$ *and* $\boldsymbol{P}_2$*, such that*

$$\boldsymbol{B} = \boldsymbol{P}_1 \begin{pmatrix} \boldsymbol{B}_1 & 0 \\ 0 & \boldsymbol{B}_2 \end{pmatrix} \boldsymbol{P}_2,$$

where $\boldsymbol{B}_1$ *and* $\boldsymbol{B}_2$ *are nested matrices.*

Proof. $1 \Rightarrow 2$: Let $\boldsymbol{B} = \operatorname{round}(\boldsymbol{l}\boldsymbol{r}^T)$. If $\boldsymbol{l}$ (or $\boldsymbol{r}$) is non-negative or non-positive, $\boldsymbol{B}$ is nested. To see this, observe that $\operatorname{round}(\boldsymbol{l}\boldsymbol{r}^T)$ remains unmodified if we replace entries of opposite sign in $\boldsymbol{r}$ (or $\boldsymbol{l}$) by 0 and then take absolute values. Then we can apply Lemma 5.15.

Otherwise, both $\boldsymbol{l}$ and $\boldsymbol{r}$ contain both strictly negative and strictly positive entries. Then there exists some permutation matrix $\boldsymbol{P}_1$, such that $\boldsymbol{P}_1^{-1}\boldsymbol{l}$ is non-increasing. We pick the vectors $\boldsymbol{l}_+ \geq 0$ and $\boldsymbol{l}_- \leq 0$, such that $\boldsymbol{P}_1^{-1}\boldsymbol{l} = \begin{pmatrix} \boldsymbol{l}_+ \\ \boldsymbol{l}_- \end{pmatrix}$. Similarly, there is some permutation matrix $\boldsymbol{P}_2$, such that $\boldsymbol{P}_2\boldsymbol{r}$ is non-increasing and we set $\boldsymbol{r}_+$ and $\boldsymbol{r}_-$ accordingly.

Using this notation we can do a quick computation,

$$\begin{aligned}
\boldsymbol{B} &= \operatorname{round}(\boldsymbol{l}\boldsymbol{r}^T) \\
&= \operatorname{round}(\boldsymbol{P}_1(\boldsymbol{P}_1^T\boldsymbol{l})(\boldsymbol{P}_2\boldsymbol{r})^T\boldsymbol{P}_2) \\
&= \boldsymbol{P}_1 \operatorname{round}\left(\begin{pmatrix} \boldsymbol{l}_+ \\ \boldsymbol{l}_- \end{pmatrix} \begin{pmatrix} \boldsymbol{r}_+ \\ \boldsymbol{r}_- \end{pmatrix}^T \right) \boldsymbol{P}_2 \\
&= \boldsymbol{P}_1 \begin{pmatrix} \operatorname{round}(\boldsymbol{l}_+\boldsymbol{r}_+^T) & \operatorname{round}(\boldsymbol{l}_+\boldsymbol{r}_-^T) \\ \operatorname{round}(\boldsymbol{l}_-\boldsymbol{r}_+^T) & \operatorname{round}(\boldsymbol{l}_-\boldsymbol{r}_-^T) \end{pmatrix}^T \boldsymbol{P}_2 \\
&= \boldsymbol{P}_1 \begin{pmatrix} \boldsymbol{B}_1 & 0 \\ 0 & \boldsymbol{B}_2 \end{pmatrix} \boldsymbol{P}_2,
\end{aligned}$$

where $\boldsymbol{B}_1 = \operatorname{round}(\boldsymbol{l}_+\boldsymbol{r}_+^T)$ and $\boldsymbol{B}_2 = \operatorname{round}(\boldsymbol{l}_-\boldsymbol{r}_-^T) = \operatorname{round}((-\boldsymbol{l}_-)(-\boldsymbol{r}_-^T))$. The last equality holds since $\operatorname{round}(\boldsymbol{l}_+\boldsymbol{r}_-^T) = \boldsymbol{0}$ and $\operatorname{round}(\boldsymbol{l}_-\boldsymbol{r}_+^T) = \boldsymbol{0}$. Finally, we observe that $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ are nested matrices by Lemma 5.15.

$2 \Rightarrow 1$: If $\boldsymbol{B}$ is nested, then $\operatorname{rrank}(\boldsymbol{B}) \leq \operatorname{rrank}_+(\boldsymbol{B}) = 1$ by Lemma 5.15. Suppose $\boldsymbol{B}$ is not nested and we are given $\boldsymbol{P}_1$, $\boldsymbol{P}_2$, $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ as in the statement of the Lemma. Then $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ are non-zero (otherwise $\boldsymbol{B}$ would be nested) and they have non-negative rounding rank one by Lemma 5.15. Thus, we can assume that $\boldsymbol{B}_1 = \operatorname{round}(\boldsymbol{l}_1\boldsymbol{r}_1^T)$ and $\boldsymbol{B}_2 = \operatorname{round}(\boldsymbol{l}_2\boldsymbol{r}_2^T)$ for some non-negative vectors $\boldsymbol{l}_1$, $\boldsymbol{l}_2$, $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$.

Now we observe that

$$\operatorname{round}\left(\begin{pmatrix}\boldsymbol{l}_1\\-\boldsymbol{l}_2\end{pmatrix}\begin{pmatrix}\boldsymbol{r}_1\\-\boldsymbol{r}_2\end{pmatrix}^T\right)=\begin{pmatrix}\boldsymbol{B}_1&0\\0&\boldsymbol{B}_2\end{pmatrix}.$$

Thus, by setting $\boldsymbol{l}=\boldsymbol{P}_1\begin{pmatrix}\boldsymbol{l}_1\\-\boldsymbol{l}_2\end{pmatrix}$ and $\boldsymbol{r}=\boldsymbol{P}_2^T\begin{pmatrix}\boldsymbol{r}_1\\-\boldsymbol{r}_2\end{pmatrix}$ we get $\boldsymbol{B}=\operatorname{round}(\boldsymbol{l}\boldsymbol{r}^T)$. □

Now we show that all identity matrices have rounding rank 2. This result was already known to Paturi and Simon [1984], where the authors picked points from the unit circle and separated them by slightly moving tangents of these points. We give another construction, since technically it appears somewhat easier.

Example 5.17. *For $n\geq 3$ let $\boldsymbol{I}_n\in\mathbb{R}^{n\times n}$ be the identity matrix. From Lemma 5.16 we get that* $\operatorname{rrank}(\boldsymbol{I}_n)>1$*, since the identity matrix is not nested. Now we look at the matrix*

$$\boldsymbol{A}=\begin{pmatrix}1&-\frac{1}{2}\\2^{-1}&-\frac{1}{2}4^{-1}\\\vdots&\vdots\\2^{-n+1}&-\frac{1}{2}4^{-n+1}\end{pmatrix}\begin{pmatrix}1&2&\dots&2^{n-1}\\1&4&\dots&4^{n-1}\end{pmatrix}=\begin{pmatrix}1-\frac{1}{2}&2-\frac{4}{2}&4-\frac{16}{2}&\cdots\\\frac{1}{2}-\frac{1}{8}&1-\frac{4}{8}&2-\frac{16}{8}&\cdots\\\frac{1}{4}-\frac{1}{32}&\frac{1}{2}-\frac{4}{32}&1-\frac{16}{32}&\cdots\\\vdots&\vdots&\vdots&\ddots\end{pmatrix}$$

and observe that $\boldsymbol{A}_{ij}=\frac{1}{2}$, if $i=j$, and $\boldsymbol{A}_{ij}<\frac{1}{2}$, otherwise. Thus, we get $\operatorname{round}(\boldsymbol{A})=\boldsymbol{I}_n$ *and therefore* $\operatorname{rrank}(\boldsymbol{I}_n)=2$.

So far we have only seen matrices with small rounding ranks. It turns out that it is rather hard to find matrices with large rounding rank. Paturi and Simon [1984] conjectured that Hadamard matrices have very large sign rank (and thus also rounding rank). This claim was proven much later by Forster [2002].

Example 5.18 ([Forster, 2002])**.** *We introduce a family of matrices, that have large sign rank. These matrices are the so called* Hadamard matrices $\boldsymbol{H}_n\in\{-1,1\}^{2^n\times 2^n}$ *and we construct them recursively. We set $\boldsymbol{H}_0=1$ and for all $n\in\mathbb{N}$, we set*

$$\boldsymbol{H}_n=\begin{pmatrix}\boldsymbol{H}_{n-1}&\boldsymbol{H}_{n-1}\\\boldsymbol{H}_{n-1}&-\boldsymbol{H}_{n-1}\end{pmatrix}.$$

Notice that the Hadamard matrices are symmetric and that the rows and columns are pairwise orthogonal (this is not hard to obtain using induction). It is well-known that $m\times n$ matrices $\boldsymbol{A}$ with pairwise orthogonal columns have operator norm $||\boldsymbol{A}||=\sqrt{m}$.

Thus applying Theorem 4.5 to $\boldsymbol{H}_n$, we get

$$\text{sign-rank}(H_n) \geq \frac{\sqrt{2^n 2^n}}{||\boldsymbol{H}_n||} = \frac{\sqrt{2^n 2^n}}{\sqrt{2^n}} = \sqrt{2^n}.$$

The above result showed that a hyperplane arrangement representing a $2^n \times 2^n$ Hadamard needs at least $2^{n/2}$ dimensions. Recently, Forster and Simon [2006] explicitly constructed a hyperplane arrangement of dimensionality $3^{n/2}$ representing H_n. Thus, their result is still worse than the $2^{n/2}$ lower bound from the previous example, but it narrows down the gap between the $2^{n/2}$ lower bound and the trivial 2^n upper bound.

Chapter 6

Comparison of the Different Ranks

This chapter is devoted to the comparison of the different ranks that we introduced in the previous chapters. We will compare the standard rank from Chapter 2, the Boolean rank from Chapter 3 and the rounding rank from Chapter 5. Sign rank will not be discussed here as we already compared it to rounding rank in Section 5.3 and since the results that we will derive for rounding rank can easily be altered to work for sign rank as well.

In each of the three sections we will compare two of the above mentioned ranks with one another. In each section we will see whether one rank serves as a bound on the other (or not). We will also discuss their domains and compare their computational complexities.

6.1 Boolean Rank and Standard Rank

Let us start by comparing the Boolean rank and the standard rank. First notice that the standard rank is defined for all real-valued matrices, while the Boolean rank's domain is only the binary matrices. Thus, the standard rank is somewhat more general. In the remainder of the section we will only consider the standard ranks of binary matrices.

Also, notice that the standard rank can be computed in cubic time (as we saw in Section 2.2), while the Boolean rank is NP-complete to compute and even hard to approximate (see Section 3.2). So, clearly, we will not be able to find exact factorisations of binary matrices with the lowest possible Boolean rank in polynomial time, unless $\mathsf{P} = \mathsf{NP}$.

Quantitatively both ranks can have large differences and neither can be treated as a bound of the other: Consider the $n \times n$ binary matrix $\boldsymbol{B}$ with $\boldsymbol{B}_{ii} = 0$, for all i, and $\boldsymbol{B}_{ij} = 1$, for $i \neq j$. Clearly, this matrix has full standard rank, i.e. $\operatorname{rank}(\boldsymbol{B}) = n$. Monson et al. [1995] prove that $\operatorname{brank}(\boldsymbol{B}) = O(\log n)$. They also give an example of matrices with Boolean rank bigger than the standard rank. Thus, neither of both ranks can be used as an upper or lower bound of the other.

6.2 Boolean Rank and Rounding Rank

This section will compare the Boolean rank and the rounding rank. Both are similar to the extent that they are only defined on the domain of binary matrices. Also, computing them is very difficult as we have seen in Section 3.2 for the Boolean rank and in Section 5.3 for the rounding rank. Hence, in both cases we will have to resort to approximate computations.

Quantitatively we can prove that the Boolean rank is an upper bound on the rounding rank of a binary matrix.

Lemma 6.1. *Let $\boldsymbol{B} \in \{0,1\}^{m \times n}$ be a binary matrix. Then its Boolean rank is an upper bound on its rounding rank, i.e.*

$$\operatorname{rrank}(\boldsymbol{B}) \leq \operatorname{brank}(\boldsymbol{B}).$$

Proof. Let $k = \operatorname{brank}(\boldsymbol{B})$. Then by definition there exist matrices $\boldsymbol{L} \in \{0,1\}^{m \times k}$ and $\boldsymbol{R} \in \{0,1\}^{n \times k}$, such that $\boldsymbol{B} = \boldsymbol{L} \circ \boldsymbol{R}^T$. Now observe that in the Boolean algebra (that is used in the Boolean matrix multiplication) we set "$1 + 1 = 1$", whereas in the algebra of the real numbers (which is used in the computation of the standard rank) we have "$1 + 1 = 2$". Thus, each entry of the matrix $\boldsymbol{L} \circ \boldsymbol{R}$ will be a lower bound on the corresponding entry of the matrix $\boldsymbol{L}\boldsymbol{R}$.

Let us define $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{R}^T$. Then for all $i = 1, \ldots, m$ and $j = 1, \ldots, m$, we have:

$$\boldsymbol{B}_{ij} = \bigvee_{l=1}^{k} (\boldsymbol{L}_{il} \wedge \boldsymbol{R}_{jl}) \leq \sum_{l=1}^{k} \boldsymbol{L}_{il}\boldsymbol{R}_{jl} = \boldsymbol{A}_{ij}.$$

Now we observe that $\operatorname{rank}(\boldsymbol{A}) \leq k$ and

$$\boldsymbol{A}_{ij} \begin{cases} \geq 1, & \text{if } \boldsymbol{B}_{ij} = 1, \\ = 0, & \text{if } \boldsymbol{B}_{ij} = 0. \end{cases}$$

Thus, we obtain $\text{round}(\boldsymbol{A}) = \boldsymbol{B}$ and, therefore, $\text{rrank}(\boldsymbol{B}) \leq k = \text{brank}(\boldsymbol{B})$. □

It is interesting to note that despite the Boolean rank being an upper bound on the rounding rank, there exist matrices with non-trivial ranks where the Boolean rank and the rounding rank coincide. One such example is the *disjointness matrix*: Consider the binary $2^n \times 2^n$ matrix $\boldsymbol{B}$ whose rows and columns are indexed by all 2^n subsets of $\{1, \ldots, n\}$. Then for $x, y \subseteq \{1, \ldots, n\}$ we set $\boldsymbol{B}_{x,y} = 1$ if and only if $|x \cap y| \geq 1$, i.e. the entry $\boldsymbol{B}_{x,y}$ contains a 1 iff x and y have at least one common element. This matrix $\boldsymbol{B}$ is known[1] to have $\text{rrank}(\boldsymbol{B}) = \text{brank}(\boldsymbol{B}) = n$.

At this point it makes sense to point out that the Boolean rank is not an upper bound on the sign rank. For example, the disjointness matrix we discussed in the previous paragraph has sign rank $n + 1$. Thus, for the sign rank we only have the upper bound $\text{sign-rank}(\boldsymbol{B}) \leq \text{rrank}(\boldsymbol{B}) + 1 \leq \text{brank}(\boldsymbol{B}) + 1$ for all binary matrices $\boldsymbol{B}$.

6.3 Rounding Rank and Standard Rank

It is left to compare the rounding rank and the standard rank of a binary matrix. Just like the Boolean rank, the rounding rank is only defined for binary matrices; thus, the standard rank has a much larger domain than the other two. Also, computationally the rounding rank is much harder to compute than the standard rank (as we saw in Sections 2.2 and 5.3).

Quantitatively it turns out that the standard rank provides an upper bound on the rounding rank.

Lemma 6.2. *Let $\boldsymbol{B} \in \{0,1\}^{m \times n}$ be a binary matrix. Then its standard rank is an upper bound on its rounding rank, i.e.*

$$\text{rrank}(\boldsymbol{B}) \leq \text{rank}(\boldsymbol{B}).$$

Proof. Let $k = \text{rank}(\boldsymbol{B})$. Then we know from Theorem 2.3 that there exist real-valued matrices $\boldsymbol{L} \in \mathbb{R}^{m \times k}$ and $\boldsymbol{R} \in \mathbb{R}^{n \times k}$ with $\boldsymbol{B} = \boldsymbol{LR}$. Now it is easy to see that using the factorisation we obtain $\boldsymbol{B} = \boldsymbol{LR} = \text{round}(\boldsymbol{LR})$ and thus $\text{rrank}(\boldsymbol{B}) \leq k = \text{rank}(\boldsymbol{B})$. □

It is interesting to observe that the rounding rank and the non-negative rounding rank from Section 5.4 will differ by at most two as we had seen in Theorem 5.13. For the

[1] The proof was given by Shay Moran in a private communication.

standard rank and the non-negative rank[2] this is not the case: Beasley and Laffey [2009] prove that the gap between the non-negative rank and the standard rank may get arbitrarily large. They show that for each $k \in \mathbb{N}$ there exists a matrix $\boldsymbol{M}_k$ with $\operatorname{rank}(\boldsymbol{M}_k) = 3$ and $\operatorname{rank}_+(\boldsymbol{M}_k) = k$.

[2]The *non-negative rank* of a matrix $\boldsymbol{M} \in \mathbb{R}^{m \times n}$, $\operatorname{rank}_+(\boldsymbol{M})$, is defined as the smallest $k \in \mathbb{N}$, such that there exist matrices $\boldsymbol{L} \in \mathbb{R}_+^{m \times k}$ and $\boldsymbol{R} \in \mathbb{R}_+^{n \times k}$ with $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{R}^T$, where $\mathbb{R}_+$ denotes the non-negative real numbers.

Chapter 7

Heuristic Algorithms for Rounding Rank

In this chapter we will see two algorithms that heuristically compute approximations of the rounding rank of a binary matrix. The first algorithm is a greedy algorithm, that uses the truncated singular value decomposition and tries to exploit the Eckart–Young Theorem. The second algorithm is a randomised algorithm that uses the geometrical interpretation of rounding rank from Theorem 5.2. We will see an empirical evaluation of both algorithms on synthetic and on real-world data in Chapter 8.

7.1 Truncated SVD Algorithm

In this section we present the *truncated SVD algorithm*, that uses truncated SVD as presented in Section 2.2 in a greedy fashion.

The algorithm works as follows: Given a binary matrix $\boldsymbol{B}$ as input, it first computes the singular value decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ of $\boldsymbol{B}$ and sets $k = 1$. Then it starts by setting $\boldsymbol{L}$ to $\boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}$ and $\boldsymbol{R}$ to $\boldsymbol{V}_{\leq k}$, i.e. $\boldsymbol{L}$ is given by the product of the first k left singular vectors and the first k singular values and $\boldsymbol{R}$ is given by the first k right singular vectors. If $\boldsymbol{L}\boldsymbol{R}^T$ rounds to the input matrix $\boldsymbol{B}$, then the algorithm returns k. Otherwise, the algorithm increases k to $k + 1$ and keeps on greedily adding singular vectors to $\boldsymbol{L}$ and $\boldsymbol{R}$ until we have $\text{round}(\boldsymbol{L}\boldsymbol{R}^T) = \boldsymbol{B}$.

The pseudocode of this procedure is given in Algorithm 1, which we will refer to as the *truncated SVD algorithm*.

Algorithm 1: Truncated SVD Algorithm

Data: A binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$.
Result: $k \in \mathbb{N}$, an approximation of the rounding rank of $\boldsymbol{B}$, and matrices $\boldsymbol{L}$ and $\boldsymbol{R}$ with $\operatorname{round}(\boldsymbol{L}\boldsymbol{R}^T) = \boldsymbol{B}$

```
Compute the SVD U Σ V^T of B
for k ← 1 to n do
    L ← U_{≤k} Σ_{≤k}
    R ← V_{≤k}
    B' ← round(L R^T)
    if B = B' then
        return k and L and R
```

Notice that the algorithm will definitely find a rounding rank decomposition: Setting $k = \operatorname{rank}(\boldsymbol{B})$ we have $\boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}\boldsymbol{V}_{\leq k}^T = \boldsymbol{B}$ and thus also $\operatorname{round}(\boldsymbol{U}_{\leq k}\boldsymbol{\Sigma}_{\leq k}\boldsymbol{V}_{\leq k}^T) = \boldsymbol{B}$. This proves the correctness of the algorithm.

The basic intutition underlying the algorithm is to exploit Theorem 2.5, which proves that the truncated SVD with k factors gives the best possible rank k approximation of any matrix in terms of the Frobenius norm. Since this approximation is globally best (i.e. after summing over *all* entries of the matrix), also the distance in each entry of the approximated matrices should be small. Thus, since each entry of the truncated SVD will be close to the real value, the rounding should deliver decent results.

This technique is not novel and was used before. For example, Erdős et al. [2014] use truncated SVD and rounding to reconstruct graphs from their neighbourhood data.

We cannot give any approximation guarantee on the algorithm, since it is possible that it performs arbitrarily bad: Consider the case that we are given the identity matrix $\boldsymbol{I}_n$ as input. On the one hand, we know from Example 5.17 that $\operatorname{rrank}(\boldsymbol{I}_n) = 2$. On the other hand, the truncated SVD with k factors will give us only k one-entries of the matrix and all other entries will be zero. Thus, for identity matrices the algorithm will always need all n factors of the truncated SVD.

7.2 Heuristic Algorithm

This section introduces the *heuristic algorithm* for computing an approximation of the rounding rank of a binary matrix. It exploits point two of the characterisation of rounding rank from Theorem 5.2. This geometric interpretation shows that computing the rounding rank is equivalent to finding points, which can be linearly separated into some given classes using affine hyperplanes. The idea of the algorithm is to randomly pick the points and then to compute the affine hyperplanes using a linear program.

We will present the algorithm in two steps: Firstly, we will give an algorithm that given a binary matrix $\boldsymbol{B}$, a dimensionality $d \in \mathbb{N}$ and a rounding threshold τ decides whether $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$. Secondly, we will use the decision algorithm from the previous step for different values of d to get an approximation of the rounding rank of the input matrix.

Decision Algorithm

In this subsection we give a randomised algorithm that decides the following problem: Given a binary matrix $\boldsymbol{B}$, a positive integer $d \in \mathbb{N}$ and a rounding threshold $\tau \in \mathbb{R}$, is $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$ true?

We start by explaining the idea behind the algorithm. In Theorem 5.2 we have seen that checking whether the rounding rank of a matrix $\boldsymbol{B}$ is at most d is equivalent to finding points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ in $\mathbb{R}^d$, which can be linearly separated into the two classes $C_j = \{\boldsymbol{l}_i : \boldsymbol{B}_{ij} = 1\}$ and $\bar{C}_j = \{\boldsymbol{l}_i : \boldsymbol{B}_{ij} = 0\}$ using affine hyperplanes. The algorithm exploits this characterisation by not directly computing a matrix $\boldsymbol{A}$ with $\operatorname{rank}(\boldsymbol{A}) \leq d$ and $\operatorname{round}(\boldsymbol{A}) = \boldsymbol{B}$, but instead by finding points and hyperplanes in $\mathbb{R}^d$, that linearly separate the points into the classes C_j and $\bar{C}_j$ as in the theorem. At the end we will construct the matrix $\boldsymbol{A}$ from these points and hyperplanes.

The description of the algorithm has three steps: Firstly, it is explained how the points are picked. Secondly, we will see the derivation of the affine hyperplanes. Thirdly, we will construct a matrix $\boldsymbol{A}$ with $\operatorname{rank}(\boldsymbol{A}) \leq d$ and $\boldsymbol{B} = \operatorname{round}(\boldsymbol{A})$.

Let us look at step one, the random choice of the points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m \in \mathbb{R}^d$. Before we see the procedure to pick the points, let us shortly consider the final situation: Eventually we will have points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ and hyperplanes $H_1, \ldots, H_n$ with the correct separation into the classes C_j and $\bar{C}_j$. Observe that those hyperplanes give us a hyperplane arrangement $\mathcal{A} = \{H_1, \ldots, H_n\}$ and that for each point $\boldsymbol{l}_i$ we will be able to compute its sign vector t_i with respect to $\mathcal{A}$. Now according to point three of Theorem 5.2 the sign vector t_i of point $\boldsymbol{l}_i$ is given by the i'th row of $\boldsymbol{B}$. Thus, in the following we will relate the choice of the points $\boldsymbol{l}_i$ to their sign vectors t_i and thus also to *their corresponding rows* in $\boldsymbol{B}$. We will denote the rows of $\boldsymbol{B}$ by $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m$.

Further observe that each point $\boldsymbol{l}_i$ identifies a region in the hyperplane arrangement and that all points in this region will have the same sign vector as $\boldsymbol{l}_i$. Thus, two regions of the hyperplane arrangement will be neighbouring if their sign vectors have Hamming distance one. On the other hand, if their sign vectors have large Hamming distance, then the regions will be separated by many hyperplanes.

Now let us get back to the situation where we want to pick the points $\boldsymbol{l}_i$. Intuitively, the algorithm picks these points randomly while trying to let them be 'close' if their corresponding rows of $\boldsymbol{B}$ have small Hamming distance.

To get a better feeling for this notion of 'closeness', consider the matrix

$$\boldsymbol{B} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

As we just argued each row of $\boldsymbol{B}$ corresponds to the sign vector of one of the points $\boldsymbol{l}_i$ and thus also to one of the regions of some hyperplane arrangement. Now looking at the regions identified by the first and the second row of $\boldsymbol{B}$, we observe that they have Hamming distance one and so they are neighbouring. Thus, when considering these regions as subsets of $\mathbb{R}^d$, then they must have points that are 'close', i.e. that have small Euclidean distance. On the other hand, a point that is in the region identified by the third row of the matrix should not be 'close' to the first two points, since the sign vectors of the regions have a large Hamming distance.

Thus, the idea of the algorithm is to find points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m \in \mathbb{R}^d$, such that if $||\boldsymbol{b}_i - \boldsymbol{b}_j||_2^2$ is small[1], then also $||\boldsymbol{l}_i - \boldsymbol{l}_j||_2^2$ is small. On the other hand, if $||\boldsymbol{b}_i - \boldsymbol{b}_j||_2^2$ is large, then there should be a reasonable distance between $\boldsymbol{l}_i$ and $\boldsymbol{l}_j$.

Now the naïve approach would be to try to find points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m \in \mathbb{R}^d$ with distances equal to the Hamming distance, i.e. with $||\boldsymbol{b}_i - \boldsymbol{b}_j||_2^2 = ||\boldsymbol{l}_i - \boldsymbol{l}_j||_2^2$ for all i and j. Unfortunately, this is not possible since this would require an isometry from $\mathbb{R}^n$ to $\mathbb{R}^d$, which does not exist for $d < n$.

Thus, we use a heuristic approach which still makes use of this intuition of closeness. We start by sampling n points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{R}^d$ uniformly at random from the unit sphere in $\mathbb{R}^d$, where the *unit sphere* S^{d-1} in $\mathbb{R}^d$ is defined as $S^{d-1} = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}||_2 = 1\}$. Now we define the points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m \in \mathbb{R}^d$ as

$$\boldsymbol{l}_i = \sum_{j=1}^{n} \boldsymbol{B}_{ij} \boldsymbol{v}_j.$$

Notice that defining the points in this way resembles the intuition of closeness we discussed before: If two rows $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ of $\boldsymbol{B}$ differ only in the k'th component, then $\boldsymbol{l}_i - \boldsymbol{l}_j = \boldsymbol{v}_k$ and thus $||\boldsymbol{b}_i - \boldsymbol{b}_j||_2^2 = ||\boldsymbol{l}_i - \boldsymbol{l}_j||_2^2 = ||\boldsymbol{v}_k||_2^2 = 1$. On the other hand, if $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ differ in

[1] Notice that since the $\boldsymbol{b}_i$ are binary, the square of the Euclidean norm of their difference coincides with their Hamming distance.

multiple components then $\boldsymbol{l}_i$ and $\boldsymbol{l}_j$ are separated by multiple vectors, since

$$\boldsymbol{l}_i - \boldsymbol{l}_j = \sum_{k=1}^{n} (\boldsymbol{B}_{ik} - \boldsymbol{B}_{jk})\boldsymbol{v}_k = \sum_{k:\boldsymbol{B}_{ik} \neq \boldsymbol{B}_{jk}} (\boldsymbol{B}_{ik} - \boldsymbol{B}_{jk})\boldsymbol{v}_k.$$

With this randomised procedure we have found some points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$. Now we continue with step two of the description of the algorithm, i.e. we will see the construction of the normal vectors $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_n$ of the hyperplanes $H_1, \ldots, H_n$, that separate the points as in Theorem 5.2. We will construct each of these vectors $\boldsymbol{r}_j$ using a linear program, that basically just states the definition of strict linear separability of the $\boldsymbol{l}_i$ into the classes C_j and $\bar{C}_j$ as given by the input matrix $\boldsymbol{B}$ and Theorem 5.2.

Notice that for the linear separability into the classes C_j and $\bar{C}_j$ as in Theorem 5.2, the $\boldsymbol{r}_j$ have to satisfy the following inequalities for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$:

$$\begin{aligned} \langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle &> \tau, \quad \text{if } \boldsymbol{B}_{ij} = 1, \\ \langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle &< \tau, \quad \text{if } \boldsymbol{B}_{ij} = 0. \end{aligned} \tag{7.1}$$

In order to derive the linear program for the computation of the $\boldsymbol{r}_j$, we observe that the constraints on the $\boldsymbol{r}_j$ are only linear. Nonetheless, we notice that linear programming does not allow for strict inequalities as we need them in Equation 7.1 for the strict linear separation. To overcome this problem, we fix a small $\varepsilon > 0$ and add it to the rounding threshold, if $\boldsymbol{B}_{ij} = 1$, or subtract it, otherwise. Notice that if the points $\boldsymbol{l}_i$ are strictly linearly separable, then in theory such an ε must exists because of the Hyperplane Separation Theorem (Theorem 5.4), but it might be very small. Hence, in practice we resort to setting ε to the smallest number bigger than zero, which can be represented by our programming language or the floating point hardware that is typically used nowadays.

Rewriting the dot products from Equation 7.1 as sums and using the insights from the above paragraph, this results in the following linear program to compute $\boldsymbol{r}_j \in \mathbb{R}^d$:

$$\begin{aligned} \min_{\boldsymbol{r}_j = (\boldsymbol{R}_{j1}, \ldots, \boldsymbol{R}_{jd})} \quad & 0 \\ \text{subject to} \quad & \sum_{k=1}^{d} \boldsymbol{L}_{ik}\boldsymbol{R}_{jk} \geq \tau + \varepsilon, \qquad \text{if } \boldsymbol{B}_{ij} = 1, \\ & \sum_{k=1}^{d} \boldsymbol{L}_{ik}\boldsymbol{R}_{jk} \leq \tau - \varepsilon, \qquad \text{if } \boldsymbol{B}_{ij} = 0. \end{aligned} \tag{7.2}$$

Observe that the objective function of the linear program is zero. This is because we do not want to optimise any objective function, but we just want to find a feasible solution satisfying the linear separability constraints.

Further notice that if this linear program has no feasible solution, then the points cannot be strictly linearly separated. In this case the algorithm will return false and stop.

The linear program from Equation 7.2 has d variables and a constraint matrix of size $m \times d$. It would also have been possible to compute all of the $\boldsymbol{r}_j$ at once, but this would require a linear program with nd variables and a constraint matrix of size $mn \times nd$, which in practice is too large to be solved efficiently due to its quadratic size compared to the smaller linear programs.

It is left to look at step three of the description of the algorithm, i.e. how we can use the $\boldsymbol{l}_i$ and the $\boldsymbol{r}_j$ in order to find a matrix $\boldsymbol{A}$ with $\operatorname{rank}(\boldsymbol{A}) \leq d$ and $\operatorname{round}_\tau(\boldsymbol{A}) = \boldsymbol{B}$. This matrix $\boldsymbol{A}$ will be a witness for $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$.

Let us define the real-valued $m \times d$ matrix $\boldsymbol{L}$ by writing the vectors $\boldsymbol{l}_i$ into its rows. Similarly, define $\boldsymbol{R} \in \mathbb{R}^{n \times d}$ by writing the $\boldsymbol{r}_j$ into its rows. Now observe that if we set $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{R}^T$, then we obtain

$$\operatorname{round}_\tau(\boldsymbol{A}) = \operatorname{round}_\tau(\langle \boldsymbol{l}_i, \boldsymbol{r}_j \rangle) = \boldsymbol{B}_{ij},$$

since the $\boldsymbol{l}_i$ and $\boldsymbol{r}_j$ satisfy the constraints from Equation 7.1. Thus, we have $\operatorname{rank}(\boldsymbol{A}) \leq d$ and $\operatorname{round}_\tau(\boldsymbol{A}) = \boldsymbol{B}$. By definition of the rounding rank this implies that we have $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$.

We have seen how to randomly pick the points $\boldsymbol{l}_i$ and how we can compute the hyperplanes given by their normal vectors $\boldsymbol{r}_j$ using linear programming. We also saw how this gives a matrix $\boldsymbol{A}$, that proves that $\operatorname{rrank}(\boldsymbol{B}) \leq d$. Thus, the description of the algorithm is finished. The pseudocode of the whole procedure is given in algorithm 2 and we will refer to it as the *randomised decision algorithm*.

Observe that the algorithm has a *one-sided error*: If it outputs *true*, then it will always be correct, since the matrices $\boldsymbol{L}$ and $\boldsymbol{R}$ serve as witnesses for the claim that $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$. However, if it outputs *false*, then this output might be a false negative. By pure chance it is possible that the guess of the points $\boldsymbol{l}_1, \ldots, \boldsymbol{l}_m$ might be unfortunate and hence they are not strictly linearly separable into the required classes, while there still exist other points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$, which satisfy this condition.

It is left to point out that there are more efficient ways to compute the hyperplanes than the linear programming approach we presented. For example, one could use Frank-Wolfe algorithms which use constrained convex methods. These methods were introduced by

Algorithm 2: A heuristic algorithm for deciding whether $\operatorname{rrank}(\boldsymbol{B}) \leq d$.

Data: A binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$, a dimensionality d and a rounding threshold $\tau \in \mathbb{R}$.

Result: True and matrices $\boldsymbol{L}$ and $\boldsymbol{R}$ with $\operatorname{round}_\tau(\boldsymbol{L}\boldsymbol{R}^T) = \boldsymbol{B}$, if the algorithm found a rounding rank decomposition in $\mathbb{R}^d$, false, otherwise.

Sample points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{R}^d$ from the unit sphere S^{d-1} uniformly at random
for $i \leftarrow 1$ *to* m **do**
    $\boldsymbol{l}_i \leftarrow \sum_{j=1}^{n} \boldsymbol{B}_{ij}\boldsymbol{v}_j$
Set $\boldsymbol{L} \leftarrow \begin{pmatrix} \boldsymbol{l}_1 \\ \vdots \\ \boldsymbol{l}_m \end{pmatrix}$
for $j \leftarrow 1$ *to* n **do**
    Construct the normal vector $\boldsymbol{r}_j$ using the linear program from equation 7.2
    if *the linear program had no feasible solution* **then**
        return *False*
Set $\boldsymbol{R} \leftarrow \begin{pmatrix} \boldsymbol{r}_1 \\ \vdots \\ \boldsymbol{r}_n \end{pmatrix}$
return *True and* $\boldsymbol{L}$ *and* $\boldsymbol{R}$

Frank and Wolfe [1956] and more recent results include the work presented in Gärtner and Jaggi [2009]. However, the linear programming approach provides the necessary functionality and while other methods would provide better running times, they do not improve the quality of the results of the algorithms.

Approximation Algorithm

Now we will take the algorithm from the previous subsection and use it to approximate the rounding rank of the input matrix. Given a binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$, a number r of runs for each dimension and a rounding threshold $\tau \in \mathbb{R}$, this algorithm will output some $d \in \mathbb{N}$ for which $\operatorname{rrank}_\tau(\boldsymbol{B}) \leq d$. It will also output matrices $\boldsymbol{L} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{R} \in \mathbb{R}^{n \times d}$, such that $\operatorname{round}_\tau(\boldsymbol{L}\boldsymbol{R}^T) = \boldsymbol{B}$.

The algorithm works by running the decision algorithm multiples times for different dimensionalities. It starts by trying to find a rounding rank decomposition of rounding rank $d = 1$. To do this, the algorithm uses the decision algorithm until it successfully finds a rounding rank decomposition of rank d or until it failed r times. In case of success, it outputs d and the obtained factorisation $\boldsymbol{L}$ and $\boldsymbol{R}$, whereas in the latter case it increases d to $d + 1$ and starts using the decision algorithm again.

The pseudocode for this procedure can be found in Algorithm 3. We will refer to this algorithm as the *heuristic algorithm.*

Algorithm 3: A heuristic algorithm to compute an approximation of $\text{rrank}_\tau(\boldsymbol{B})$.

Data: A binary matrix $\boldsymbol{B} \in \{0,1\}^{m \times n}$, a $r \in \mathbb{N}$ as the number of runs for each dimensionality and a rounding threshold $\tau \in \mathbb{R}$.

Result: $d \in \mathbb{N}$ as approximation of $\text{rrank}_\tau(\boldsymbol{B})$ and matrices $\boldsymbol{L}$ and $\boldsymbol{R}$ with $\boldsymbol{B} = \text{round}_\tau(\boldsymbol{L}\boldsymbol{R}^T)$

```
1 for d ≥ 1 do
2     for run ← 1 to r do
3         (success, L, R) ← run Algorithm 2 with inputs B, d and τ
4         if success then
5             return d and L and R
```

The reason for starting the decision algorithm r times for each value of d is that the decision algorithm randomly picks the matrix $\boldsymbol{L}$. By starting the decision algorithm multiple times we increase the probability to obtain a matrix $\boldsymbol{L}$ with points, that are strictly linearly separable into the classes given by the matrix $\boldsymbol{B}$. Thus, running this algorithm multiple times increases the probability to obtain a correct result.

As a possible improvement to decrease the running time of the heuristic algorithm one might want to determine the correct dimensionality d by using a version of binary search, instead of increasing d one by one. In theory this would minimise the workload needed to find the minimal d. In practice this turned out to be very slow, since it took a long time to solve the linear programs for large values of d, that were never considered when increasing d one by one (since in practice the rounding ranks of most matrices turned out to be rather small as we will see in Chapter 8). Nonetheless, we notice that there certainly exist more efficient strategies for increasing the values of d.

Chapter 8

Experiments

Both algorithms that were presented in Chapter 7 were implemented in Matlab and tested on both synthetic and on real-world data. In this chapter we will first look at the data the algorithms were tested on and then evaluate their results.

8.1 Test Data

This section is devoted to the the data that was used to test the algorithms. It will first be argued why it is difficult to find data on which one can run the algorithms to give a meaningful evaluation. After that we will see how matrices can be generated for which we know an upper bound on their rounding ranks. Finally, we will see descriptions of the real-world data sets that were used.

To evaluate the quality of the algorithms we presented, we need matrices for which we know their exact rounding ranks. Unfortunately, there exist only very few matrices for which we know their *exact* ranks. We had seen some examples in Section 5.5, but most of those matrices had only very small rounding ranks or we only knew a lower bound on their ranks like for the Hadamard matrices. This problem could be solved by implementing a naïve algorithm, that is slow, but works on small matrices and computes their rounding ranks exactly. Unfortunately, to the author's best knowledge no such naïve algorithm exists.

To test the algorithms on matrices of 'medium' and 'high' ranks one option would be to just randomly sample binary matrices. Then by Theorem 4.7 we know that those matrices will have a large rounding rank with high probability. Nonetheless, we do not know what their exact rounding ranks are. So, when running the algorithms on those

matrices, one could only compare the results of the algorithms to the theoretically known lower bounds on their rounding ranks.

Hence, the algorithms were tested on identity matrices (for which we know that their rounding rank is two from Example 5.17) and on heuristically created data.

Let us the discuss the heuristic creation of synthetic data. The idea is to sample two real-valued factor matrices with k factors, multiply them and then to round them. Then, clearly, the resulting matrix must have a rounding rank of at most k. More formally, for a given number of factors, $k \in \mathbb{N}$, real-valued matrices $\boldsymbol{L} \in \left[-\frac{1}{2}, \frac{1}{2}\right]^{n \times k}$ and $\boldsymbol{R} \in \left[-\frac{1}{2}, \frac{1}{2}\right]^{k \times n}$ were sampled uniformly at random. Then we know that for the resulting binary matrix $\boldsymbol{B} = \operatorname{round}_0(\boldsymbol{LR})$, we must have $\operatorname{rrank}(\boldsymbol{B}) \leq k$. We still do not know what the exact rounding rank of $\boldsymbol{B}$ is, but we have an upper bound. Due to the results from Theorem 4.7, for small k this upper bound should be almost tight and even for larger values of k the gap should not be too large.

The algorithms were also tested on real-world data and the following two paragraphs will provide a short overview on how those data sets were fetched. The *Abstracts* data set[1] is a collection of project abstracts that were submitted to the National Science Foundation of the USA in applications for funding (for the preprocessing of the data see Miettinen [2009, page 84]). It contains 12841 abstracts and 4894 words, where entry (i, j) of the corresponding matrix contains a one iff the i'th abstract contains the j'th word. The *DBLP* data[2] was fetched from the famous DBLP website. It contains information about 6980 authors and their publications at 19 computer science conferences. An entry contains a one iff an author published at the conferences. The *NOW* data set[3] has size 124×139 and contains information about at which locations the fossils of certain species were found. It was fetched by Fortelius [2003] and preprocessed according to Fortelius et al. [2006]. In the *Dialects* data from Embleton and Wheeler [1997] and Embleton and Wheeler [2000], linguists matched 1334 features of Finnish dialects to 506 different Finnish municipalities. More detailed information about these data sets can be found in Miettinen [2009, section 4.8.4], which was also used to provide the above descriptions.

The algorithms were also tested on data from Hewlett-Packard that was used by Ene et al. [2008]. These data sets contain network access control rules and are called *americas large*, *americas small*, *apj*, *emea*, *healthcare*, *domino*, *customer*, *firewall 1* and *firewall 2*. Their dimensionalities can be found in Table 8.3 on page 67. Note that from Ene et al. [2008] we know the exact Boolean ranks of these data sets and this paper also provides more information on what information they contain.

[1] `http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html`
[2] `http://dblp.uni-trier.de/db/`
[3] `http://www.helsinki.fi/science/now/`

8.2 Evaluation of the Algorithms

This section will provide the results the algorithms achieved on the data we described in the previous section. We will discuss their results on both synthetic and on real-world data and we will further describe the dependency of the results of the algorithms on their parameters.

The author wishes the reader to notice that we are only comparing the algorithms that were presented in Chapter 7. We will not compare against the algorithms mentioned in Section 4.3, as the author did not consider those algorithms as implementable.

Evaluation on Synthetic Data

Let us start by looking at how the results of the heuristic algorithm depend on the parameter r, i.e. on the number of runs it performs per dimension.

The heuristic algorithm was run on synthetic matrices of size 500×400 with parameter $r = 1, 10, 100$. The matrices were created with different numbers of factors as described in the previous section. The outcomes of these experiments are given in Table 8.1 and the results are also visualised in the plot given in Figure 8.1.

The results from Table 8.1 show that the the algorithm's outputs were very similar for the different values of r. In absolute numbers, the rounding rank approximations for $r = 1$ and $r = 100$ differed by at most six, which relatively is less than 5% of a difference

TABLE 8.1: Results of experiments to investigate how much the heuristic algorithm improves with more runs per dimensionality. The first column gives the type of the data set, m is its number of rows, n is its number of columns. Column four gives the standard rank of the data set and column five gives the number of factors that were used for the data generation. The last three columns give the output of the heuristic algorithm for different values of parameter r. There average over the algorithm's outputs is given by μ and σ states the standard deviation of these outputs.

Data	m	n	rank	factors	μ	σ	heuristic		
							$r = 1$	$r = 10$	$r = 100$
synthetic	500	400	400	25	100.3	2.31	103	99	99
				50	120.3	2.31	123	119	119
				75	132.3	2.08	134	133	130
				100	141.3	3.21	145	140	139
				125	147	1.73	148	148	145
				150	150.3	0.58	150	151	150
				175	156.3	2.52	159	156	154
				200	157.7	1.53	159	158	156

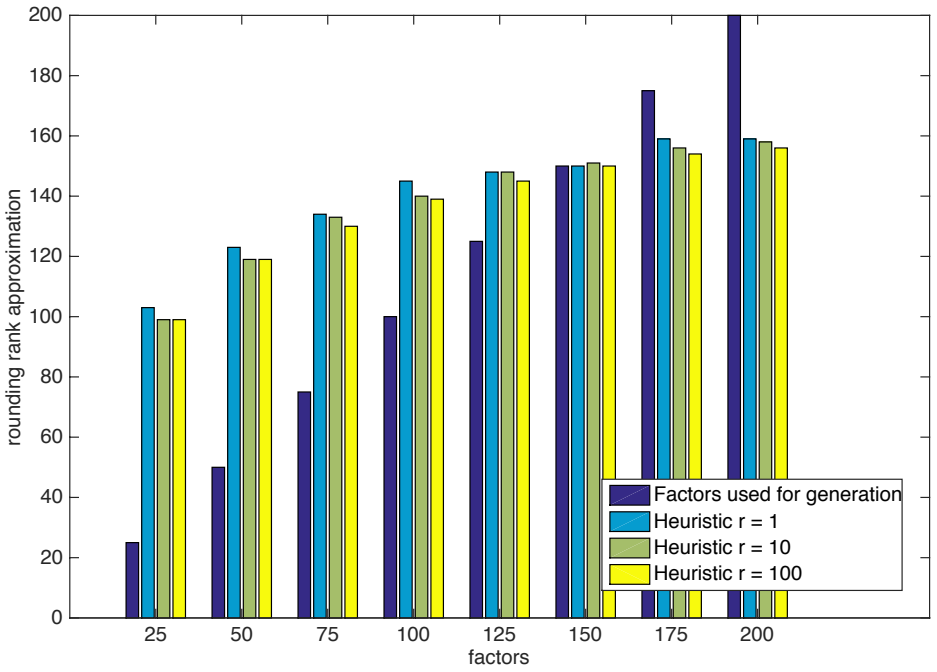

FIGURE 8.1: Results of the heuristic algorithm with parameters $r = 1, 10, 100$ on synthetically created binary matrices of size 500×400. The x-axis gives the number of factors that were used for the generation of the matrices. This figure visualises the results from Table 8.1.

in the quality of the algorithms. The small standard deviations from column seven of the table further strengthen this interpretation. Thus, we can observe that larger parameters for r do increase the accuracy of the algorithm, but only slightly.

This allows the interpretation that the probabilities that the randomised decision algorithm (Algorithm 2) gets a correct answer are very small until at a certain point they start to increase dramatically. The author did not run further tests to back up this claim, but this intuition goes along well with other experiments he ran.

Due to the observation that the heuristic algorithm only improves slightly with increasing values of r, all further experiments were run with fixed parameter $r = 1$ in order to decrease the running times of the algorithm.

To compare the truncated SVD algorithm and the heuristic algorithm, let us look at the results that are given in Table 8.2.

The results presented in the table show that as argued in Section 7.1 the truncated SVD algorithm performs very badly on the identity matrices; it needs all singular vectors to find a matrix that rounds to the identity matrix. The heuristic algorithm does very well and even for large $10^4 \times 10^4$ identity matrices it outputs three, where the optimal solution would be two.

TABLE 8.2: Results of both algorithms on synthetic data. The first column gives the type of the data set, m is its number of rows, n is its number of columns. Column four gives the standard rank of the data set and column five gives the exact rounding rank (if it is known). In case of synthetic data, the sixth column gives the number of factors that were used for the data generation. The last two columns give the outputs of the algorithms. The heuristic algorithm was run with parameter $r = 1$.

Data	m	n	rank	rrank	factors	SVD	heuristic
identity matrix	10^2	10^2	10^2	2	–	10^2	2
	10^3	10^3	10^3	2	–	10^3	3
	10^4	10^4	10^4	2	–	10^4	3
synthetic	1000	800	800	–	25	412	161
				–	75	461	217
				–	125	442	248
				–	175	452	270
				–	225	472	280
				–	275	465	289
				–	325	475	299

Next, let us the discuss the results of the algorithms on the synthetic data from Table 8.2. Seven different binary 1000×800 matrices were created with different numbers of factors as discussed in the previous section. All of the synthetically created matrices had full standard rank. Those results are visualised in Figure 8.2.

We observe that for the synthetic data, both the truncated SVD algorithm and also the heuristic algorithm provided much better results than the naïve upper bound provided by the standard rank. Also, it is evident that the heuristic algorithm delivered much more accurate results than the truncated SVD algorithm, as often it was better by a factor of two. Particularly for matrices that were created with a small number of factors its results were much better.

For the truncated SVD algorithm it is evident that although it delivered results which were a factor two better than the rounding rank upper bound given by the standard rank, it returned almost constant approximations of the rounding ranks of the matrices. This indicates that it cannot utilise the structure of the matrices. One cause might be that despite the fact that truncated SVD delivers the best possible low-rank approximations of a given matrix in the Frobenius norm (according to the Eckart–Young Theorem), it only minimises this global property. The rounding rank, on the other hand, has a more local focus, since it is concerned about being on the right side of the rounding threshold in each entry.

Concerning the development of the rounding rank approximations in terms of the number of factors we used to generate the matrices, let us make three observations.

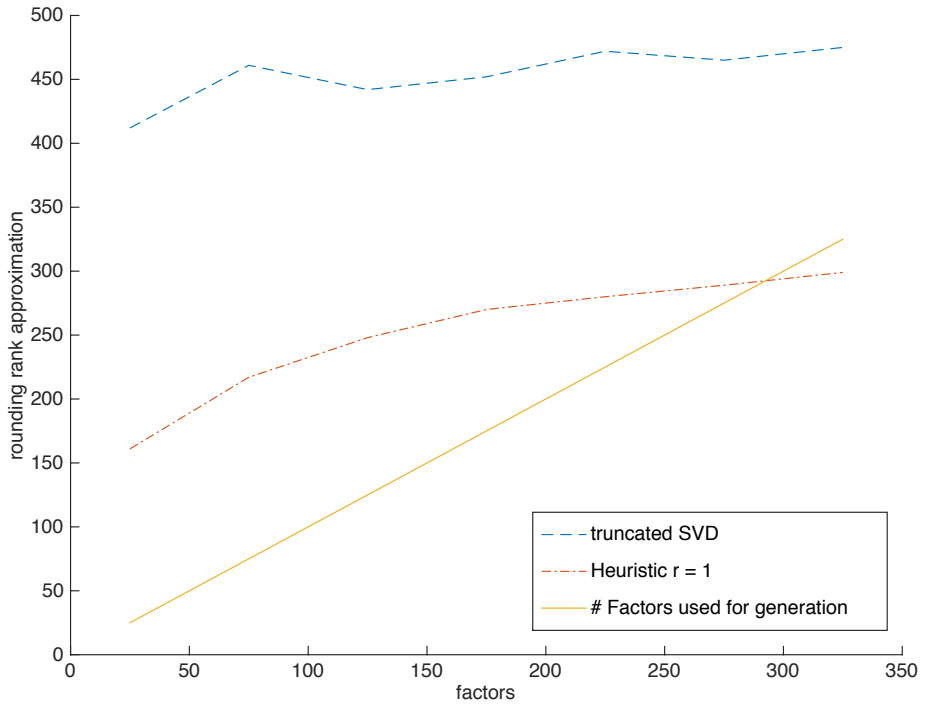

FIGURE 8.2: Results of the truncated SVD algorithm (blue dashed line) and the heuristic with $r = 1$ (orange line with dashes and dots) on synthetically created binary matrices of size 1000×800. The x-axis gives the number of factors that were used for the generation of the matrices, which are also given by the yellow line. This figure visualises the numbers from Table 8.2.

Firstly, let us notice that although the number of factors used for the generation of the of the data was raised from 25 to 325, the results from the truncated SVD algorithm were always between 412 and 475. Also, the rounding rank approximations that were obtained are not monotonic in the number of factors used, e.g. for the matrix with 75 factors it output 461 and for the one with 125 factors it output 442. This shows that the approximations from this algorithm are rather poor, since they do not seem to correspond to the ground truth.

Secondly, observe that the approximations from the heuristic algorithm do show the just mentioned monotonicity, i.e. the more factors were used for the generation of the matrices, the larger the approximation was. This shows that the algorithm indeed seems to be able to exploit the structure of the matrices.

Thirdly, let us discuss the quality of algorithms' approximations. It is interesting to see that for the matrix generated with 325 factors the approximation from the heuristic algorithm 'outperformed' the data generation process and output a factorisation of rank 299. This behaviour might be attributed to the fact that the upper bound from the synthetic data generation gets too loose. On the other hand, for matrices that had very

small rounding ranks the approximations from both algorithms were not completely convincing. For the matrix created with 25 factors, the heuristic algorithm's result was off by a factor of 6 and the truncated SVD algorithm's result was off by a factor of 16. Nonetheless, when increasing the number of factors the quality of the results seemed to improve. Of course, this might also be caused by a too loose upper bound in the data generation process.

Evaluation on Real-World Data

Let us finish the evaluation of the algorithms by discussing their results on real-world data sets. Table 8.3 contains their outputs on the real-world data sets we introduced in the previous section.

Again, we can observe that both algorithms delivered better results than the upper bound given by the standard rank. Once more the heuristic algorithm was much better than the truncated SVD algorithm, which in three cases was only slightly better than the trivial approximations given by the standard rank.

Looking at the results from the heuristic algorithm, we see that all data sets appear to have much smaller rounding ranks than standard ranks. Particularly for the larger data

TABLE 8.3: Results of the algorithms run on real-world data. For the last nine data sets we their exact Boolean ranks were provided by Ene et al. [2008]. The first column gives the name of the data set, m is its number of rows, n its number of columns. Columns four and five give the standard and the Boolean ranks of the data sets. The second-last column gives the output of the truncated SVD algorithm and the last one gives the heuristic algorithm's output (it was run with parameter $r = 1$).

Data	m	n	rank	brank	SVD	heuristic
NOW	124	139	123	–	68	25
DBLP	19	6980	19	–	19	11
Dialects	1334	506	506	–	445	87
Abstracts	12841	4894	4893	–	4421	451
americas small	3477	1587	203	178	185	28
americas large	3485	10127	404	398	386	47
apj	2044	1164	455	453	443	24
emea	35	3046	34	34	34	12
healthcare	46	46	14	14	11	5
domino	79	231	20	20	17	6
customer	1021	277	276	276	271	59
firewall 1	365	709	68	64	58	15
firewall 2	325	590	10	10	9	4

sets it is intriguing to observe that their rounding ranks seem to be less than 20% of their standard ranks.

For the data sets taken from Ene et al. [2008] we can also compare their rounding ranks to their (exact) Boolean ranks. Notice the intriguing fact that for all data sets the Boolean rank never got larger than the standard rank and that in four out of nine cases it was even smaller. This is somewhat surprising since we had seen in Section 6.1 that in general this is not the case.

With the results from the algorithms we can once more observe that the heuristic algorithm provides much better bounds on the rounding rank than the truncated SVD algorithm. Also, it is highly interesting that in many cases the heuristic algorithm provided rounding ranks that are at most 15% of the standard ranks. For the *apj* data set the approximation of its rounding rank is almost only 5% of its standard rank.

It is left to discuss the running times of the algorithms. Since the implementations were not supposed to be as efficient as possible, but were only meant to provide good qualitative results, it was refrained from giving a quantitative analysis of the running times. Instead, the following rough observations are provided: Both algorithms always finished running within at most twenty minutes and on most instances in much less than five minutes. The only exception is the *Abstracts* data set, which is the largest data set that was considered, where the heuristic algorithm needed one week to find a solution, while the truncated SVD algorithm terminated after one hour. In all cases the truncated SVD algorithm finished faster than the heuristic algorithm. One reason for this is the fact that Matlab's SVD implementation is highly optimised and thus very efficient. Also, the heuristic algorithm's computation of the hyperplanes was done using linear programming. If it had been implemented using Frank-Wolfe algorithms as pointed out in Section 7.2, the heuristic algorithm could have been sped up significantly.

Chapter 9

Conclusion

This thesis introduced the rounding rank of a binary matrix and investigated it both mathematically and also experimentally.

We saw the rounding rank compared to the well-known standard rank and the Boolean rank. We also gave an extensive summary of the literature on sign rank and observed how similar it is to the rounding rank.

The rounding rank was also characterised as problem of combinatorial geometry and also as one of embedding points in Euclidean spaces. These equivalent problems helped us to prove that the rounding threshold can change the rounding rank of a given binary matrix by at most one.

The thesis further presented two algorithms to approximate the rounding rank. The experiments on both synthetic and on real-world data sets showed that while the rounding rank of a matrix is NP-hard to compute, many matrices had rather small rounding ranks. Particularly for the real-world data sets we were able to observe that it had only very little rounding ranks.

In the introduction of the thesis we posted the question 'How powerful is the rounding rank?'. We argued that if it is very powerful, then it should always be much smaller than the standard rank or the Boolean rank of a binary matrix. On the other hand, if those ranks are very similar, then it is not very powerful.

In the course of thesis we saw in Chapter 6 that the rounding rank gives a lower bound on the standard rank and on the Boolean rank. Nonetheless, we observed in Section 4.4 that there are matrices for which there is no asymptotical difference between their standard ranks and their rounding ranks. Hence, theoretically the rounding rank is not more powerful than the standard rank in the worst case.

In spite of that, the experiments from Chapter 8 indicated that in practise the rounding rank is much smaller than the standard rank of a matrix. Particularly, we were able to observe that real-world data sets tend to have rather small rounding ranks. Hence, in applications the rounding seems to be an interesting and powerful technique.

Let us finish the thesis by discussing what future directions of research there might be.

There seems to be an interesting way to improve the heuristic algorithm from Section 7.2. Looking at the decision algorithm, one might interpret the random selection of the points as a random projection of the input matrix. Thus, it might be insightful to analyse the algorithm in the light of the Johnson–Lindenstrauss lemma (see Johnson and Lindenstrauss [1984]). Also, one might want to use the results from Achlioptas [2003] to obtain a better random project than the one used in the algorithm, which should also be more efficient to compute as it will provide sparser matrices.

In terms of data analysis one might want to find the best low-rounding-rank approximation of a given binary matrix. In other words, while the algorithms presented in this thesis compute an exact reconstruction of the input matrix, such an approach would allow for errors in the reconstruction. Indeed, the methods from both presented algorithms can easily be altered to allow such approximations. The truncated SVD algorithm could simply stop adding singular vectors at a certain point. The heuristic algorithm could be updated in the following way: Instead of computing the hyperplanes exactly, we could introduce slack variables as in soft margin support vector machines (see, e.g. Bishop [2006]) in order to classify as many points correctly as possible.

It would also be interesting to further investigate the matrices, that we obtain from the heuristic algorithm. For example, one might want to take a closer look at which entries of the matrices are close to the rounding threshold and which ones are not. If these entries are (approximately) always the same, then we could argue that for a binary matrix consisting of multiple classification problems, these entries will be hard to classify (i.e. close to the decision boundaries) by linear classifiers independently of the input data. It might also be interesting to visualise these points and to check if we can observe some patterns.

Bibliography

Achlioptas, D. (2003). Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671 – 687.

Achlioptas, D. and McSherry, F. (2001). Fast computation of low rank matrix approximations. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 611–618, New York, NY, USA. ACM.

Alon, N., Frankl, P., and Rodl, V. (1985). Geometrical realization of set systems and probabilistic communication complexity. In *26th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 277–280.

Alon, N., Haussler, D., and Welzl, E. (1987). Partitioning and geometric embedding of range spaces of finite Vapnik–Chervonenkis dimension. In *Proceedings of the Third Annual Symposium on Computational Geometry*, SCG '87, pages 331–340, New York, NY, USA. ACM.

Alon, N., Moran, S., and Yehudayoff, A. (2014). Sign rank, VC dimension and spectral gaps. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:135.

Basri, R., Felzenszwalb, P., Girshick, R., Jacobs, D., and Klivans, C. (2009). Visibility constraints on features of 3D objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1231–1238.

Beasley, L. B. and Laffey, T. J. (2009). Real rank versus nonnegative rank. *Linear Algebra and its Applications*, 431(12):2330 – 2335.

Ben-David, S., Eiron, N., and Simon, H. (2001). Limitations of learning via embeddings in euclidean half-spaces. In *Computational Learning Theory*, volume 2111 of *Lecture Notes in Computer Science*, pages 385–401. Springer Berlin Heidelberg.

Bhangale, A. and Kopparty, S. (2015). The complexity of computing the minimum rank of a sign pattern matrix. *Computing Research Repository (CoRR)*, abs/1503.04486.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Blumer, A., Ehrenfeucht, A., Haussler, D., and Warmuth, M. K. (1989). Learnability and the Vapnik–Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York, NY, USA.

Canny, J. F. (1988). Some algebraic and geometric computations in pspace. Technical Report UCB/CSD-88-439, EECS Department, University of California, Berkeley.

Chalermsook, P., Heydrich, S., Holm, E., and Karrenbauer, A. (2014). Nearly tight approximability results for minimum biclique cover and partition. In *Algorithms - ESA 2014*, volume 8737 of *Lecture Notes in Computer Science*, pages 235–246. Springer Berlin Heidelberg.

Drineas, P., Fountoulakis, K., and Kundu, A. (2015). A randomized rounding algorithm for sparse PCA. *Computing Research Repository (CoRR)*, abs/1508.03337.

Edelsbrunner, H. (1987). *Algorithms in Combinatorial Geometry*. Springer-Verlag New York, Inc., New York, NY, USA.

Embleton, S. and Wheeler, E. S. (1997). Finnish dialect atlas for quantitative studies. *Journal of Quantitative Linguistics*, 4(1-3):99–102.

Embleton, S. M. and Wheeler, E. S. (2000). Computerized dialect atlas of finnish: Dealing with ambiguity. *Journal of Quantitative Linguistics*, 7(3):227–231.

Ene, A., Horne, W., Milosavljevic, N., Rao, P., Schreiber, R., and Tarjan, R. E. (2008). Fast exact and heuristic methods for role minimization problems. In *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies*, SACMAT '08, pages 1–10, New York, NY, USA. ACM.

Erdős, D., Gemulla, R., and Terzi, E. (2014). Reconstructing graphs from neighborhood data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(4):23:1–23:22.

Forster, J. (2002). A linear lower bound on the unbounded error probabilistic communication complexity. *Journal of Computer and System Sciences*, 65(4):612 – 625.

Forster, J., Krause, M., Lokam, S., Mubarakzjanov, R., Schmitt, N., and Simon, H. (2001). Relations between communication complexity, linear arrangements, and computational complexity. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science*, volume 2245 of *Lecture Notes in Computer Science*, pages 171–182. Springer Berlin Heidelberg.

Forster, J., Schmitt, N., Simon, H., and Suttorp, T. (2003). Estimating the optimal margins of embeddings in euclidean half spaces. *Machine Learning*, 51(3):263–281.

Forster, J. and Simon, H. U. (2006). On the smallest possible dimension and the largest possible margin of linear arrangements representing given concept classes. *Theoretical Computer Science*, 350(1):40 – 48.

Fortelius, M., Gionis, A., Jernvall, J., and Mannila, H. (2006). Spectral ordering and biochronology of european fossil mammals. *Paleobiology*, 32(2):206–214.

Fortelius, M. c. (2003). New and old worlds database of fossil mammals (NOW). Online. `http://www.helsinki.fi/science/now/`. Accessed: 2015-09-23.

Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Gärtner, B. and Jaggi, M. (2009). Coresets for polytope distance. In *Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry*, SCG '09, pages 33–42, New York, NY, USA. ACM.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition.

Johnson, W. B. and Lindenstrauss, J. (1984). Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1–1.

Junttila, E. (2011). *Patterns in permuted binary matrices*. PhD thesis, University of Helsinki.

Kang, R. and Müller, T. (2012). Sphere and dot product representations of graphs. *Discrete & Computational Geometry*, 47(3):548–568.

Kearns, M. J. and Vazirani, U. V. (1994). *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, USA.

Krause, M. (1991). Geometric arguments yield better bounds for threshold circuits and distributed computing. In *Proceedings of the Sixth Annual Structure in Complexity Theory Conference*, pages 314–321.

Leskovec, J., Rajaraman, A., and Ullman, J. D. (2014). *Mining of massive datasets.* Cambridge University Press.

Linial, N., Mendelson, S., Schechtman, G., and Shraibman, A. (2007). Complexity measures of sign matrices. In *In Proceedings of the 39th ACM Symposium on the Theory of Computing. ACM.*

Linial, N. and Shraibman, A. (2008). Learning complexity vs. communication complexity. In *23rd Annual IEEE Conference on Computational Complexity*, pages 53–63.

Matousek, J. (2014). Intersection graphs of segments and $\exists\mathbb{R}$. *Computing Research Repository (CoRR)*, abs/1406.2636.

Miettinen, P. (2009). *Matrix decomposition methods for data mining: Computational complexity and algorithms.* PhD thesis, Helsingin yliopisto.

Mnev, N. (1985). Realizability of combinatorial types of convex polyhedra over fields. *Journal of Soviet Mathematics*, 28(4):606–609.

Mnev, N. (1988). The universality theorems on the classification problem of configuration varieties and convex polytopes varieties. In *Topology and Geometry — Rohlin Seminar*, volume 1346 of *Lecture Notes in Mathematics*, pages 527–543. Springer Berlin Heidelberg.

Monson, S. D., Pullman, N. J., and Rees, R. (1995). A survey of clique and biclique coverings and factorizations of (0; 1)-matrices. *Bulletin of the ICA*, 14:17 – 86.

Orlik, P. and Terao, H. (1992). *Arrangements of Hyperplanes.* Grundlehren der mathematischen Wissenschaften : a series of comprehensive studies in mathematics. Springer.

Paige, C. C. (1981). Properties of numerical algorithms related to computing controllability. *IEEE Transactions on Automatic Control*, 26(1):130–138.

Paturi, R. and Simon, J. (1984). Probabilistic communication complexity. In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science (FOCS)*, SFCS '84, pages 118–126, Washington, DC, USA. IEEE Computer Society.

Razborov, A. and Sherstov, A. (2008). The sign-rank of AC^o. In *IEEE 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 57–66.

Rudin, W. (1987). *Real and Complex Analysis, 3rd Ed.* McGraw-Hill, Inc., New York, NY, USA.

Shor, P. (1991). Stretchability of pseudolines is NP-hard. *Applied Geometry and Discrete Mathematics, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 4:531–554.

Simon, H. U. (1990). On approximate solutions for combinatorial optimization problems. *SIAM Journal on Discrete Mathematics*, 3(2):294–310.

Stanley, R. P. (2004). An introduction to hyperplane arrangements. In *Lecture notes, IAS/Park City Mathematics Institute*.

Trevisan, L. (2004). Inapproximability of combinatorial optimization problems. *The Computing Research Repository*.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

Vapnik, V. N. and Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.

Ziegler, G. (1995). *Lectures on Polytopes*. Graduate Texts in Mathematics. Springer New York.

Ziegler, G. M. (2002). Oriented matroids. *The Electronic Journal of Combinatorics*, 1000:DS4–Sep.

Appendix A

Appendix

A.1 Collection of Definitions and Lemmata

This appendix offers two definitions and a lemma, which are well-known. They are stated here for the sake of completeness.

Definition A.1 (Linear Separability, see [Boyd and Vandenberghe, 2004, section 2.5])**.** *Let X_0 and X_1 be two sets of points in $\mathbb{R}^n$.*

Then X_0 and X_1 are linearly separable *if there exists a vector $\boldsymbol{v} \in \mathbb{R}^n$ and a real number $\theta \in \mathbb{R}$, such that for every point $\boldsymbol{x} \in X_0$ we have $\langle \boldsymbol{x}, \boldsymbol{v} \rangle \geq \theta$ and every point $\boldsymbol{y} \in X_1$ satisfies $\langle \boldsymbol{y}, \boldsymbol{v} \rangle \leq \theta$.*

X_0 and X_1 are strictly linearly separable *if there exists a vector $\boldsymbol{v} \in \mathbb{R}^n$ and a number $\theta \in \mathbb{R}$, such that for every point $\boldsymbol{x} \in X_0$ we have $\langle \boldsymbol{x}, \boldsymbol{v} \rangle > \theta$ and every point $\boldsymbol{y} \in X_1$ satisfies $\langle \boldsymbol{y}, \boldsymbol{v} \rangle < \theta$.*

Lemma A.2 ([Boyd and Vandenberghe, 2004, page 423])**.** *Let A and B be finite sets of points in $\mathbb{R}^d$. Then A and B are (strictly) linearly separable if and only if their convex hulls are (strictly) linearly separable.*

Definition A.3 ([Golub and Van Loan, 1996, page 109])**.** *A square binary matrix $\boldsymbol{P} \in \{0,1\}^{n \times n}$ is called a* permutation matrix*, if we can reorder the rows of $\boldsymbol{P}$ to obtain the $n \times n$ identity matrix.*