Algorithms in Discrepancy Theory and Lattices

Harishchandra Ramadas

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2017

Reading Committee:

Thomas Rothvoß, Chair

Dmitriy Drusvyatskiy

Rekha Thomas

Simge Küçükyavuz (GSR)

Program Authorized to Offer Degree:
Mathematics

University of Washington

Abstract

Algorithms in Discrepancy Theory and Lattices

Harishchandra Ramadas

Chair of the Supervisory Committee:
Assistant Professor Thomas Rothvoß
Department of Mathematics

This thesis deals with algorithmic problems in discrepancy theory and lattices, and is based on two projects I worked on while at the University of Washington in Seattle. A brief overview is provided in Chapter 1 (Introduction).

Chapter 2 covers joint work with Avi Levy and Thomas Rothvoß in the field of discrepancy minimization [28]. A well-known theorem of Spencer shows that any set system with n sets over n elements admits a coloring of discrepancy $O(\sqrt{n})$. While the original proof was non-constructive, recent progress brought polynomial time algorithms by Bansal, Lovett and Meka, and Rothvoß. All those algorithms are randomized, even though Bansal's algorithm admitted a complicated derandomization. We propose an elegant deterministic polynomial time algorithm that is inspired by Lovett-Meka as well as the Multiplicative Weight Update method. The algorithm iteratively updates a fractional coloring while controlling the exponential weights that are assigned to the set constraints. A conjecture by Meka suggests that Spencer's bound can be generalized to symmetric matrices. We prove that $n \times n$ matrices that are block diagonal with block size q admit a coloring of discrepancy $O(\sqrt{n} \cdot \sqrt{\log(q)})$. Bansal, Dadush and Garg recently gave a randomized algorithm to find a vector x with entries in $\{-1, 1\}$ with $\|Ax\|_\infty \leq O(\sqrt{\log n})$ in polynomial time, where A is any matrix whose columns have length at most 1. We show that our method can be used to deterministically obtain such a vector.

In Chapter 3, we discuss a result in the broad area of lattices and integer optimization, in joint work with Rebecca Hoberg, Thomas Rothvoß and Xin Yang [22]. The number balancing (NBP) problem is the following: given real numbers $a_1, \ldots, a_n \in [0, 1]$, find two disjoint subsets $I_1, I_2 \subseteq [n]$ so that the difference $|\sum_{i \in I_1} a_i - \sum_{i \in I_2} a_i|$ of their sums is minimized. An application of the pigeonhole principle shows that there is always a solution where the difference is at most $O(\frac{\sqrt{n}}{2^n})$. Finding the minimum, however, is NP-hard. In polynomial time, the *differencing algorithm* by Karmarkar and Karp from 1982 can produce a solution with difference at most $n^{-\Theta(\log n)}$, but no further improvement has been made since then. We show a relationship between NBP and Minkowski's Theorem. First we show that an approximate oracle for Minkowski's Theorem gives an approximate NBP oracle. Perhaps more surprisingly, we show that an approximate NBP oracle gives an approximate Minkowski oracle. In particular, we prove that any polynomial time algorithm that guarantees a solution of difference at most $2^{\sqrt{n}}/2^n$ would give a polynomial approximation for Minkowski as well as a polynomial factor approximation algorithm for the Shortest Vector Problem.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I want to thank the following people for helping make this thesis possible:

- My advisor, Thomas Rothvoß, for all the help, guidance and support in the course of my Ph.D.

- My co-authors — Rebecca Hoberg, Avi Levy, Thomas Rothvoß and Xin Yang — for being great collaborators, and for their permission to use content from our papers in this thesis.

- Thomas Rothvoß, Dima Drusvyatskiy, Rekha Thomas and Simge Küçükyavuz for serving on my committee.

- My colleagues and teachers at the University of Washington for many useful lessons and discussions.

- My family and friends.

Chapter 1

INTRODUCTION

1.1 Deterministic Discrepancy Minimization

Discrepancy minimization is a classical area in combinatorial optimization that has attracted a great deal of interest in recent years [6, 7, 31, 45] . The *multiplicative weights update* method has numerous applications in algorithm design, game theory and machine learning [4]. We use this method to construct efficient deterministic algorithms to solve classical problems in discrepancy theory.

1.1.1 A Constructive Version of Spencer's Theorem

Given a collection of n elements and a system of m subsets of this collection, our goal is to assign to each element one of two colors, represented by $+1$ and -1, so that each subset has approximately the same number of elements colored $+1$ and -1.

More formally, let $S_1, \ldots, S_m$ be subsets of $[n] = \{1, \ldots, n\}$. We want to find a mapping

$$\chi : [n] \to \{-1, 1\},$$

so that

$$\max_{i \in [m]} \left| \sum_{j \in S_i} \chi(j) \right|$$

is minimized. If we call $\left|\sum_{j \in S_i} \chi(j)\right|$ the *discrepancy* of S_i, our goal is to find a coloring χ that minimizes the maximum discrepancy of the family. We call

$$\min_{\chi : [n] \to \{-1,1\}} \max_{i \in [m]} \left| \sum_{j \in S_i} \chi(j) \right|$$

the *discrepancy of the family* $S_1, \ldots, S_m$.

In the special case where the size of the set system equals the number of elements (i.e. $m = n$), it is not hard to find a coloring χ that gives a discrepancy of $O\left(\sqrt{n \log n}\right)$. In fact, a *random* coloring does the job: if we pick the color of each element independently by flipping an unbiased coin, then standard concentration inequalities (e.g. Chernoff bounds) show that the discrepancy of a fixed set S_i is at most $O\left(\sqrt{n \log n}\right)$ with probability $1 - o\left(\frac{1}{n}\right)$. This probability is large enough that we can easily take a union bound over the n subsets to show that with high probability, a random coloring does indeed give a discrepancy of $O\left(\sqrt{n \log n}\right)$ for the entire set system.

On the other hand, it is not hard to show that there exists a set system $S_1, \ldots, S_n$ over n elements such that the discrepancy is $\Omega(\sqrt{n})$. Such a system can be constructed explicitly; a randomized construction where each subset is defined by picking every element $j \in [n]$ independently with probability $1/2$ also suffices.

Bridging the gap between the $\sqrt{n \log n}$ upper bound and $\sqrt{n}$ lower bound remained open until 1985, when Spencer [48] published his groundbreaking paper *Six Standard Deviations Suffice*. This paper showed that in fact, there always exists a coloring χ whose discrepancy is $O(\sqrt{n})$—specifically, $6\sqrt{n}$, hence the title of the paper. However, his result was nonconstructive; the proof of the existence of such a coloring relied on a pigeonhole principle argument with exponentially many pigeons and holes.

A polynomial-time algorithm to obtain an $O(\sqrt{n})$-discrepancy coloring remained elusive for the next 25 years. In his breakthrough work in 2010, Bansal [6] presented a polynomial-time randomized algorithm based on a semidefinite programming (SDP)-based relaxation of the original problem. (The algorithm was later derandomized in [8].)

Shortly afterwards, in 2012, Lovett and Meka [31] presented an efficient randomized algorithm that finds a low-discrepancy *partial coloring*: a point $x \in [-1, 1]^n$ with $\Omega(n)$ many coordinates equal to $+1$ or -1. Let K be the polytope defined by the (linear) inequalities specifying that each subset has discrepancy at most $O(\sqrt{n})$. Their algorithm found a partial coloring by running a random walk in the polytope $[-1, 1]^n \cap K$; on hitting a face of the polytope, the walk continues walking within that face by choosing increments orthogonal to

the normal to the face. By iteratively applying the algorithm $\log n$ times, it is easy to obtain a *full* coloring in $\{-1,1\}^n$ with $O(\sqrt{n})$ discrepancy. Unlike Bansal, Lovett and Meka were able to show that their algorithm produced a low-discrepancy coloring without requiring existential arguments.

In 2014, Rothvoß [45] generalized these results even further, by presenting a simple algorithm that produces a point in $[-1,1]^n \cap K$ with $\Omega(n)$ many coordinates equal to $+1$ or -1, where K is *any* convex set with Gaussian measure at least $e^{-O(n)}$. Crucially, K is not necessarily required to be a polytope; it suffices to have a polynomial-time separation oracle for K.

Our Contribution: We present a deterministic polynomial-time algorithm to obtain an $O(\sqrt{n})$-discrepancy coloring for a set system of size $O(n)$ over n variables. We do this by innovatively applying the *multiplicative weights update method* to obtain a partial coloring under identical conditions as the theorem of Lovett and Meka. Formally, let $v_1, \ldots, v_m$ be unit vectors in $\mathbb{R}^n$ and $\lambda_1, \ldots, \lambda_m$ satisfy

$$\sum_{i=1}^{m} e^{-\lambda_i^2} \leq O(n).$$

Then we have a deterministic polynomial-time algorithm that finds a vector $x \in [-1,1]^n$ with

$$|\{j \in [n] : |x_j| = 1\}| \geq \Omega(n),$$

and $\langle x, v_i \rangle \leq \lambda_i$ for every $i \in [m]$.

To find the vector x, the algorithm executes a deterministic walk inside the hypercube $[-1,1]^n$, freezing a coordinate j to either $+1$ or -1 if the walker hits the surface $x_j = 1$ or $x_j = -1$ respectively. In other words, once the walker reaches a face of the hypercube, it continues its walk *within* that face. The size of each step of the walker is bounded below in most iterations, and each step is chosen to be orthogonal to the position vector of the current location of the walker. This ensures that distance from the origin increases after every step; since the walker never leaves the hypercube, this distance is bounded, ensuring that the total

number of steps is at most $\text{poly}(n, \frac{1}{\delta^2})$, where $\delta = 1/\text{poly}(n)$ is the step size. Each iteration itself takes time $\text{poly}(n, m)$, because of which the algorithm as a whole is polynomial time.

For each constraint $i \in [m]$, we associate a *weight*, w_i that changes every iteration. If the walker walks in the direction of the hyperplane defining a constraint, the corresponding weight increases by a multiplicative factor; if the walker walks away from the hyperplane, the weight decreases. If the weight associated with a constraint is extremely high, this indicates that the walker is in danger of crossing the associated hyperplane, and thus leaving the polytope defined by the intersection of the hypercube and the halfspaces corresponding to the discrepancy constraints. In such a situation, the walker simply walks orthogonal to this hyperplane; we show that there are at most $O(n)$ such heavy weights during any given iteration.

Importantly, we define a *potential function* that is simply the sum of weights in a given iteration. By carefully choosing the update step, the algorithm is designed to ensure that with each iteration in the walk, the potential function does not increase. This ensures that *on average*, the weights do not increase. As described in the previous paragraph, by walking orthogonal to the constraints with especially heavy weights, we ensure that in fact each individual weight is bounded. Since for every constraint $i \in [m]$, the associated weight is bounded, the constraint is not violated; the discrepancy of the corresponding subset is at most $O(\sqrt{n})$. Finally, we run the algorithm until at least $n/2$ coordinates are frozen, giving us a low-discrepancy partial coloring.

1.1.2 The Beck-Fiala Setting and Banaszczyk's Theorem

A special case of the discrepancy problem is the so-called *Beck-Fiala* or *sparse set* setting. As before, we are given subsets $S_1, \ldots, S_m$ of $[n]$, and our goal is to obtain a coloring

$$\chi : [n] \to \{-1, 1\},$$

whose discrepancy is as small as possible. However, in this setting, we restrict our attention to set systems where each element $j \in [n]$ is in at most s subsets, for some $s > 0$.

A simple linear algebraic argument given by Beck and Fiala in 1981 [10] showed that in this setting, one can find a coloring with discrepancy at most $2s - 1 = O(s)$. They conjectured, however, that the discrepancy is in fact $O(\sqrt{s})$.

In 1998, Banaszczyk [5] proved the existence of an $O\left(\sqrt{s \log n}\right)$-discrepancy coloring for the Beck-Fiala setting. He showed that for *any* symmetric convex set K with Gaussian measure at least $\Omega(1)$ and matrix $A \in \mathbb{R}^{m \times n}$ with columns having l^2-norm at most $O(1)$, there exists a vector $x \in \{-1, 1\}^n$ with $Ax \in K$. To get an $O\left(\sqrt{s \log n}\right)$-discrepancy coloring in the Beck-Fiala setting, we let row i of the matrix A be $1/\sqrt{s}$ times the indicator vector of the set $i \in [m]$ and let K be the cube $\left[-\sqrt{\log n}, \sqrt{\log n}\right]^n$ and apply Banaszczyk's theorem. These results, however, were all nonconstructive.

Bansal's 2010 paper [6] gave the first constructive proof for the existence of an $O(\sqrt{s} \log n)$-discrepancy coloring in the Beck-Fiala setting. In 2016, Bansal, Dadush and Garg [7] eliminated a $\sqrt{\log n}$-factor by giving a polynomial-time randomized algorithm that produces a coloring with $O\left(\sqrt{s \log n}\right)$ discrepancy. Their algorithm is based on a random walk in a polytope, and requires a semidefinite program (SDP) to be solved in every iteration to determine the next step in the walk. Unlike previous approaches, their algorithm did not require iteratively applying a partial coloring lemma; instead, they ensured that progress was made in coloring elements in *every* subset in each iteration.

Our Contribution: We give a polynomial-time deterministic algorithm based on the multiplicative weights update method that produces an $O\left(\sqrt{s \log n}\right)$-discrepancy coloring in the Beck-Fiala setting. More generally, given an $m \times n$ matrix A whose columns have l^2 norm at most 1, our algorithm outputs a vector $x \in \{-1, 1\}^n$ with

$$\|Ax\|_\infty \leq O\left(\sqrt{\log n}\right).$$

This is a special case of Banasczyk's theorem, where the only convex set we consider is a hypercube of side length $\Omega\left(\sqrt{\log n}\right)$. (It can be easily verified that this hypercube has Gaussian measure $\Omega(1)$, if the constants are chosen properly, so the hypotheses of the theorem are satisfied.) The algorithm in [7] outputs such a vector in randomized polynomial

time. Our algorithm does so deterministically, and unlike the algorithm in [7], does not require an SDP to be solved in every step of the walk.

It can also be seen that the desired coloring in the Beck-Fiala setting can be easily produced by our algorithm. The matrix A that we choose has columns indexed by elements $j \in [n]$ and rows indexed by subsets $i \in [m]$. We set the element $A_{ij} := 1$ if element j is contained in subset i and 0 otherwise. Since by assumption each element is contained in at most s subsets, the l^2-norm of each column is at most $\sqrt{s}$. Scaling the matrix down by a factor of $\sqrt{s}$ gives a matrix with columns having l^2-norm at most 1; we then apply our algorithm to this matrix to output a vector $x \in \{-1, 1\}^n$. Since the discrepancy for the scaled matrix is $O\left(\sqrt{\log n}\right)$, the original matrix has discrepancy $O\left(\sqrt{s \log n}\right)$.

Our algorithm is based on a deterministic walk in the polytope obtained by intersecting the cube $[-1, 1]^n$ with the polytope defined by the discrepancy constraints. As in the basic setting – when there was no condition on the number of subsets containing each element – we define a potential function that we then ensure never increases. The algorithm penalizes the walker if a constraint comes close to being violated; however, the walker is also rewarded for making progress coloring the elements in each individual set. As a consequence, we ensure that we make progress coloring *every* subset.

1.1.3 Discrepancy Theory for Matrices

For a matrix $A \in \mathbb{R}^{n \times n}$, we use $\|A\|_{\text{op}}$ to denote its *operator norm*:

$$\|A\|_{\text{op}} := \max_{\|x\|_2 = 1} \|Ax\|_2.$$

Let $A_1, \ldots, A_n$ be symmetric $n \times n$ matrices with $\|A_i\|_{\text{op}} \leq 1$ for every $i \in [n]$. Our goal is to pick a vector of signs $x \in \{-1, 1\}^n$ so that $\| \sum_{i=1}^n x_i A_i \|_{\text{op}}$ is minimized. An application of a matrix Chernoff bound shows that for a *random* assignment of signs $x \in \{-1, 1\}^n$, we have $\| \sum_{i=1}^n x_i A_i \|_{\text{op}} \leq O\left(\sqrt{n \log n}\right)$ with high probability.

In fact, there is a deterministic algorithm based on the multiplicative weights update

method that produces $x \in \{-1,1\}^n$ so that

$$A := \left\| \sum_{i=1}^{n} x_i A_i \right\|_{\text{op}} \leq O\left(\sqrt{n \log n}\right).$$

The algorithm picks $x_1, x_2, \ldots$ sequentially so that the trace of the exponential of the matrix A is bounded in every iteration; we do this using the *Golden-Thompson inequality*[1]. Bounding the trace of the (positive semidefinite) matrix exponential then gives a trivial upper bound on each individual eigenvalue, and hence the matrix operator norm.

If, however, the matrices A_i are diagonal (or simultaneously diagonalizable), an application of Spencer's theorem shows that there exists $x \in \{-1,1\}^n$ so that

$$\left\| \sum_{i=1}^{n} x_i A_i \right\|_{\text{op}} \leq O\left(\sqrt{n}\right).$$

In 2014, Meka [37] conjectured that this should be true even when the matrices are *not* diagonalizable; this has not been proven (or disproved) till date.

For an integer q between 1 and n, we call a family of $n \times n$ matrices $A_1, \ldots, A_n$ *q-block diagonal* if the matrices are simultaneously block diagonal with blocks of size at most $q \times q$.

Our Contribution: For a family $A_1, \ldots, A_n$ of $n \times n$ symmetric q-block diagonal matrices with $\|A_i\|_{\text{op}} \leq 1$ for all i, we have a deterministic polynomial-time algorithm based on the multiplicative weights update method that produces $x \in \{-1,1\}^n$ so that

$$\left\| \sum_{i=1}^{n} x_i A_i \right\|_{\text{op}} \leq O\left(\sqrt{n \log q}\right).$$

The algorithm, as before, is based on a deterministic walk in the hypercube $[-1,1]^n$. The potential function is written as the sum of separate potential functions for each of the $q \times q$ blocks. These in turn are written as the trace of the exponential of a matrix; as before, we pick our update vector from a carefully chosen subspace and use the Golden-Thompson inequality to bound the increase in the potential function.

[1]The Golden-Thompson inequality says that for symmetric matrices A and B, $\text{Tr}(e^{A+B}) \leq \text{Tr}(e^A \cdot e^B)$ (see, for example, [11]).

This result can be thought of as interpolating between the previously known $O\left(\sqrt{n \log n}\right)$ discrepancy and Meka's conjectured $O\left(\sqrt{n}\right)$ bound, and potentially represents progress in proving the full bound.

1.2 Minkowski's Theorem and Number Partitioning are Comparable

The *number balancing problem* is a variant of the *number partitioning problem*, a classical problem in theoretical computer science with numerous practical applications [38]. The goal is to partition a given set of numbers into two subsets, such that the sums of the numbers in each subset are approximately equal. (These two subsets should not both be empty; however, there may be elements that are in neither set.) The elements in the first subset can be thought of as being assigned minus $(-)$ signs and those in the second subset plus $(+)$ signs; the goal then is to minimize the absolute value of the signed sum. Formally speaking, given a vector of numbers $a = (a_1, \ldots, a_n) \in [0,1]^n$, we want to find a vector $x \in \{-1,0,1\}^n \setminus \{0\}$ so that $|\langle x, a\rangle| = |\sum_{i=1}^n x_i a_i|$ is minimized.

A simple application of the pigeonhole principle can be used to show that there exists an $x \in \{-1,0,1\}^n \setminus \{0\}$ so that $|\langle x, a\rangle| \leq \frac{n}{2^n-1}$. A slightly refined argument gives us an improved bound of $|\langle x, a\rangle| \leq O\left(\frac{\sqrt{n}}{2^n}\right)$.

It is well-known, however, that finding a vector of signs x that minimizes $|\langle x, a\rangle|$ is NP-hard. The best-known algorithm for the number partitioning problem that runs in polynomial time is Karmarkar and Karp's *differencing algorithm* from 1986 [24], that produces a vector $x \in \{-1,0,1\}^n \setminus \{0\}$ that satisfies $|\langle x, a\rangle| \leq n^{-\Theta(\log(n))}$.

Minkowski's theorem, proved by Hermann Minkowski in 1889, is a fundamental result from lattices and integer optimization. The simplest version states that if $K \subset \mathbb{R}^n$ is a symmetric convex set with $\mathrm{vol}(K) > 2^n$, then K must contain at least one integer point other than the origin, i.e. $K \cap (\mathbb{Z}^n \setminus \{0\}) \neq \emptyset$ [36].

The proof of Minkowski's theorem is elementary but nonconstructive; it relies on an argument involving overlap between translated copies of the set $K/2$, and the number of such copies we consider scales exponentially with the dimension n. Given a symmetric convex set K with $\mathrm{vol}(K) > 2^n$, actually *finding* a nonzero integer point in K is believed to be much harder. It is known that even an approximate constructive version of Minkowski's theorem would provide an approximation algorithm for the *shortest vector problem* for lattices, which

in turn could be used to break lattice-based cryptosystems [2, 40].

Our Contribution: We show that an approximate oracle for Minkowski's theorem implies a good bound for the number partitioning problem, and that an approximate oracle for the number partitioning problem implies a good constructive approximation for Minkowski's theorem.

1.2.1 Reducing Number Balancing to Minkowski's Theorem

If we have a polynomial-time *exact* oracle for Minkowski's theorem, obtaining a polynomial time algorithm for the number balancing problem with an exponentially small bound is not very hard. Given a vector of numbers

$$a := (a_1, \ldots, a_n) \in [0,1]^n,$$

one can show using elementary convex geometry that for some $\delta = e^{-\Theta(n)}$, the polytope

$$(-2,2)^n \cap \{x : |\langle a, x\rangle| \leq \delta\}$$

has volume greater than 2^n. The Minkowski oracle can then be used to find a nonzero integer point in this polytope, which must have entries in $\{-1, 0, 1\}$. This basic approach can be made to work with $(2-\epsilon)$-approximate oracles, for any $\epsilon > 0$, but cannot be pushed any further.

If all we have is a ρ-approximate oracle for some $\rho \geq 2$, we can employ the same approach to obtain a vector x with entries in

$$\{-k, -(k-1), \ldots, (k-1), k\}$$

for some integer $k \geq 2$ so that $\langle a, x\rangle \leq \delta$. We present a self-reduction that uses this to obtain a vector x with entries in

$$\{-k/2, -(k/2-1), \ldots, (k/2-1), k/2\}.$$

(For the sake of notational simplicity, we assume for the moment that k is a multiple of 2.)

The self-reduction proceeds as follows: we divide the set of numbers $(a_1, \ldots, a_n)$ into subsets of size $\sqrt{n}$. For a subset l, we find a number $b_l \neq 0$ so that each of $b_l, 2b_l, \ldots, kb_l$ can be expressed as a linear combination of numbers in subset l with coefficients only in

$$\{-k/2, -(k/2-1), \ldots, (k/2-1), k/2\}.$$

We then use our oracle on the numbers $b_1, \ldots, b_{\sqrt{n}}$ to find a $y \in \{-k, \ldots, k\}^{\sqrt{n}}$ with

$$\langle b, y \rangle = \sum_{l=1}^{\sqrt{n}} b_l y_l$$

small. Since each summand can be expressed as a linear combination of $(a_1, \ldots a_n)$ with coefficients in

$$\{-k/2, -(k/2-1), \ldots, (k/2-1), k/2\},$$

we obtain a solution x with entries in this range.

1.2.2 Reducing Minkowski's Theorem to Number Balancing

Given a polynomial-time oracle for number balancing, we present a polynomial-time algorithm for a constructive approximate version of Minkowski's theorem. The basic idea is described in the following paragraphs.

First, any symmetric convex body K can be approximated up to a small polynomial factor by an ellipsoid, a result known as *John's theorem*. If K is a polytope, this ellipsoid approximates K up to a factor of $\sqrt{n}$ and can be computed in time polynomial in n and the number of facets. (If K is given by a separation oracle, we can only approximate K with an ellipsoid up to a factor of n in polynomial time.) We may hence assume that the body K is an ellipsoid.

Next, we may also assume that the ellipsoid we consider is *well-rounded*, which means that the lengths of the axes are bounded. We show that this follows from *lattice basis reduction*, a process where we find a unimodular map that transforms the ellipsoid to a well-rounded one while mapping the integer lattice $\mathbb{Z}^n$ to itself. Since unimodular maps preserve volumes, the transformed ellipsoid has the same volume as the original one.

We show that given a number balancing oracle with output smaller than some factor δ, we can apply this oracle *simultaneously* on all the axes of the well-rounded ellipsoid to obtain a nonzero point $x \in \mathbb{Z}^n$ that has small inner product with the vectors defining each of the axes. Since all the inner products are sufficiently small, x must lie inside the ellipsoid, upto a polynomial scaling factor.

Chapter 2

DETERMINISTIC DISCREPANCY MINIMIZATION

2.1 Introduction

The classical setting in (combinatorial) *discrepancy theory* is that a set system $S_1, \ldots, S_m \subseteq \{1, \ldots, n\}$ over a ground set of n elements is given and the goal is to find *bi-coloring* $\chi : \{1, \ldots, n\} \to \{\pm 1\}$ so that the worst imbalance $\max_{i=1,\ldots,m} |\chi(S_i)|$ of a set is minimized. Here we abbreviate $\chi(S_i) := \sum_{j \in S_i} \chi(j)$.

A seminal result of Spencer [48] says that there is always a coloring χ where the imbalance is at most $O\left(\sqrt{n \cdot \log(2m/n)}\right)$ for $m \geq n$. The proof of Spencer is based on the *partial coloring method* that was first used by Beck in 1981 [9]. The argument applies the *pigeonhole principle* to obtain that many of the 2^n many colorings χ, χ' must satisfy

$$|\chi(S_i) - \chi'(S_i)| \leq O\left(\sqrt{n \cdot \log(2m/n)}\right)$$

for all sets S_i. Then one can take the *difference* between such a pair of colorings with $|\{j \mid \chi(j) \neq \chi'(j)\}| \geq \frac{n}{2}$ to obtain a *partial coloring* of low discrepancy. This partial coloring can be used to color half of the elements. Then one iterates the argument and again finds a partial coloring. As the remaining set system has only half the elements, the bound in the second iteration becomes better by a constant factor. This process is repeated until all elements are colored; the total discrepancy is then given by a convergent series with value $O\left(\sqrt{n \cdot \log(2m/n)}\right)$. More general arguments based on convex geometry were given by Gluskin [19] and by Giannopoulos [18], but their arguments still relied on a pigeonhole principle with exponentially many pigeons and pigeonholes and did not lead to polynomial time algorithms.

In fact, Alon and Spencer [3] even conjectured that finding a coloring satisfying Spencer's

theorem would be intractable. In a breakthrough, Bansal [6] showed that one could set up a *semi-definite program* (SDP) to find at least a vector coloring, using Spencer's Theorem to argue that the SDP has to be feasible. He then argued that a random walk guided by updated solutions to that SDP would find a coloring of discrepancy $O(\sqrt{n})$ in the balanced case $m = n$. However, his approach needed a very careful choice of parameters.

A simpler and truly constructive approach that does not rely on Spencer's argument was provided by Lovett and Meka [31], who showed that for $x^{(0)} \in [-1, 1]^n$, any polytope of the form

$$P = \left\{x \in [-1, 1]^n : \left|\left\langle v_i, x - x^{(0)}\right\rangle\right| \leq \lambda_i \; \forall i \in [m]\right\}$$

contains a point that has at least half of the coordinates in $\{-1, 1\}$. Here it is important that the polytope P is large enough; if the normal vectors v_i are scaled to unit length, then the argument requires that

$$\sum_{i=1}^{m} e^{-\lambda_i^2/16} \leq \frac{n}{16}.$$

Their algorithm surprisingly simple: start a Brownian motion at $x^{(0)}$ and stay inside any face that is hit at any time. They showed that this random walk eventually reaches a point with the desired properties.

More recently, the Rothvoß [45] provided another algorithm which simply consists of taking a random Gaussian vector x and then computing the nearest point to x in P. In contrast to both of the previous algorithms, this argument extends to the case that $P = Q \cap [-1, 1]^n$ where Q is any symmetric convex set with a large enough Gaussian measure.

However, all three algorithms described above are randomized, although Bansal and Spencer [8] could derandomize the original arguments by Bansal. They showed that the random walk already works if the directions are chosen from a 4-wise independent distribution, which then allows a polynomial time derandomization.

In our algorithm, we think of the process more as a *multiplicative weight update* procedure, where each constraint has a weight that increases if the current point moves in the direction of its normal vector. The potential function we consider is the sum of those weights. Then

in each step we simply need to select an update direction in which the potential function does not increase.

The multiplicative weight update method is a meta-algorithm that originated in game theory but has found numerous recent applications in theoretical computer science and machine learning. In the general setting one imagines having a set of experts (in our case the set constraints) that are assigned an exponential weight that reflects the value of the gain/loss that expert's decisions had in previous rounds. Then in each iteration one selects an update, which can be a convex combination of experts, where the convex coefficient is proportional to the current weight of the expert[1]. We refer to the very readable survey of Arora, Hazan and Kale [4] for a detailed discussion.

2.1.1 Related work

If we have a set system $S_1, \ldots, S_m$ where each element lies in at most t sets, then the partial coloring technique described above can be used to find a coloring of discrepancy $O\left(\sqrt{t} \cdot \log n\right)$ [50]. A linear programming approach of Beck and Fiala [10] showed that the discrepancy is bounded by $2t-1$, independent of the size of the set system. On the other hand, there is a non-constructive approach of Banaszczyk [5] that provides a bound of $O\left(\sqrt{t \log n}\right)$ using convex geometry arguments. Only very recently, a corresponding algorithmic bound was found by Bansal, Dadush and Garg [7]. A conjecture of Beck and Fiala says that the correct bound should be $O\left(\sqrt{t}\right)$. This bound can be achieved for the vector coloring version, see Nikolov [43].

More generally, the theorem of Banaszczyk [5] shows that for any convex set K with Gaussian measure at least $\frac{1}{2}$ and any set of vectors $v_1, \ldots, v_m$ of length $\|v_i\|_2 \leq \frac{1}{5}$, there exist signs $\varepsilon_i \in \{\pm 1\}$ so that $\sum_{i=1}^{m} \varepsilon_i v_i \in K$.

A set of k permutations on n symbols induces a set system with kn sets given by the prefix intervals. One can use the partial coloring method to find a $O(\sqrt{k} \log n)$ discrepancy

[1]We should mention for the sake of completeness that our update choice is *not* a convex combination of the experts weighted by their exponential weights.

coloring [49], while a linear programming approach gives a $O(k \log n)$ discrepancy [13]. In fact, for any k one can always color half of the elements with a discrepancy of $O(\sqrt{k})$ — this even holds for each induced sub-system [49]. Still, [41] constructed 3 permutations requiring a discrepancy of $\Theta(\log n)$ to color all elements.

Also the recent proof of the Kadison-Singer conjecture by Marcus, Spielman and Srivastava [33] can be seen as a discrepancy result. They show that a set of vectors $v_1, \ldots, v_m \in \mathbb{R}^n$ with $\sum_{i=1}^m v_i v_i^T = I$ can be partitioned into two halves S_1, S_2 so that

$$\sum_{i \in S_j} v_i v_i^T \preceq \left(\frac{1}{2} + O\left(\sqrt{\varepsilon}\right)\right) I$$

for $j \in \{1, 2\}$ where $\varepsilon = \max_{i=1,\ldots,m}\{\|v_i\|_2^2\}$ and I is the $n \times n$ identity matrix. Their method is based on interlacing polynomials; no polynomial time algorithm is known to find the desired partition.

For a symmetric matrix $A \in \mathbb{R}^{m \times m}$, let $\|A\|_{\text{op}}$ denote the largest singular value; in other words, the largest absolute value of any eigenvalue. The discrepancy question can be generalized from sets to symmetric matrices $A_1, \ldots, A_n \in \mathbb{R}^{m \times m}$ with $\|A_i\|_{\text{op}} \leq 1$ by defining

$$\text{disc}(\{A_1, \ldots, A_n\}) := \min\left\{\left\|\sum_{i=1}^n x_i A_i\right\|_{\text{op}} : x \in \{-1, 1\}^n\right\}.$$

Note that picking $0/1$ diagonal matrices

$$A_i := \text{diag}(1_{S_i})$$

would exactly encode the set coloring setting. Again the interesting case is $m = n$; in contrast to the diagonal case it is only known that the discrepancy is bounded by $O\left(\sqrt{n \cdot \log(n)}\right)$, which is already attained by a random coloring. Meka[2] conjectured that the discrepancy of n matrices can be bounded by $O(\sqrt{n})$.

For a very readable introduction into discrepancy theory, we recommend Chapter 4 in the book of Matoušek [35] or the book of Chazelle [14].

---

[2]See the blog post
`https://windowsontheory.org/2014/02/07/discrepancy-and-beating-the-union-bound/`.

2.1.2 Our contribution

Our main result is a deterministic version of the theorem of Lovett and Meka:

Theorem 1. *Let $v_1, \ldots, v_m \in \mathbb{R}^n$ unit vectors, $x^{(0)} \in [-1,1]^n$ be a starting point and let $\lambda_1 \geq \ldots \geq \lambda_m \geq 0$ be parameters so that*

$$\sum_{i=1}^{m} \exp(-\lambda_i^2/16) \leq \frac{n}{32}.$$

Then there is a deterministic algorithm that computes a vector $x \in [-1,1]^n$ with

$$\langle v_i, x - x_0 \rangle \leq 11\lambda_i$$

for all $i \in [m]$ and $|\{i : x_i = \pm 1\}| \geq \frac{n}{2}$, in time $O(\min\{n^4 m, n^3 m \lambda_1^2\})$.

By setting $\lambda_i = O(1)$ this yields a deterministic version of Spencer's theorem in the balanced case $m = n$:

Corollary 2. *Given n sets over n elements, there is a deterministic algorithm that finds a $O(\sqrt{n})$-discrepancy coloring in time $O(n^4)$.*

Furthermore, Spencer's *hyperbolic cosine algorithm* [47] can also be interpreted as a multiplicative weight update argument. However, the techniques of [47] are only enough for a $O\left(\sqrt{n \log(n)}\right)$ discrepancy bound for the balanced case. Our hope is that similar arguments can be applied to solve open problems such as whether there is an extension of Spencer's result to balance matrices [52] and to better discrepancy minimization techniques in the Beck-Fiala setting. To demonstrate the versatility of our arguments, we show an extension to the matrix discrepancy case.

We say that a symmetric matrix $A \in \mathbb{R}^{m \times m}$ is *q-block diagonal* if it can be written as

$$A = \operatorname{diag}(B_1, \ldots, B_{m/q}),$$

where each B_j is a symmetric $q \times q$ matrix.

Theorem 3. *For given q-block diagonal matrices $A_1, \ldots, A_n \in \mathbb{R}^{m \times m}$ with $\|A_i\|_{op} \leq 1$ for $i = 1, \ldots, n$ one can compute a coloring $x \in \{-1, 1\}^n$ with*

$$\left\| \sum_{i=1}^{n} x_i A_i \right\|_{op} \leq O\left(\sqrt{n \log\left(\frac{2qm}{n} \right)} \right)$$

deterministically in time $O(n^5 + n^4 m^3)$.

Finally, we can also give the first deterministic algorithm for the result of Bansal, Dadush and Garg [7].

Theorem 4. *Let $A \in \mathbb{R}^{m \times n}$ be a matrix with $\|A^j\|_2 \leq 1$ for all columns $j = 1, \ldots, n$. Then there is a deterministic algorithm to find a coloring $x \in \{-1, 1\}^n$ with*

$$\|Ax\|_\infty \leq O\left(\sqrt{\log n} \right)$$

in time $O(n^3 \log(n) \cdot (m + n))$.

While [7] need to solve a semidefinite program in each step of their random walk, our algorithm does not require solving any SDPs. Note that we do not optimize running times such as by using fast matrix multiplication.

In the Beck-Fiala setting, we are given a set system over n elements, where each element is contained in at most t subsets. Theorem 4 then provides the first polynomial-time deterministic algorithm that produces a coloring with discrepancy $O(\sqrt{t \log n})$; we simply choose the matrix A whose rows are the incidence vectors of members of the set system, scaled by $1/\sqrt{t}$.

We present the proof of Theorem 3 in Section 2.4.

2.2 The algorithm for partial coloring

We will now describe the algorithm proving Theorem 1. First note that for any $\lambda_i > 2\sqrt{n}$ we can remove the constraint $\langle v_i, x - x_0 \rangle \leq \lambda_i$, as it does not cut off any point in $[-1, 1]^n$. Thus we assume without loss of generality that

$$2\sqrt{n} \geq \lambda_1 \geq \cdots \geq \lambda_m \geq 0.$$

Let

$$\delta := \frac{1}{\lambda_1}$$

denote the step size of our algorithm. The algorithm will run for $O(n/\delta^2)$ iterations, each of computational cost $O(n^2 m)$. Note that $\delta = O(1/\sqrt{n})$ so the algorithm terminates in $O(n^2)$ iterations. The total runtime is hence

$$O(n^2 m \cdot n/\delta^2) = O(n^3 m \lambda_1^2) \leq O(n^4 m).$$

For a symmetric matrix $M \in \mathbb{R}^{n \times n}$ we know that an *eigendecomposition*

$$M = \sum_{j=1}^{n} \mu_j u_j u_j^T$$

can be computed in time $O(n^3)$. Here

$$\mu_j := \mu_j(M)$$

is the *jth eigenvalue* of M and

$$u_j := u_j(M)$$

is the corresponding *eigenvector* with $\|u_j\|_2 = 1$. We make the convention that the eigenvalues are sorted as $\mu_1 \geq \ldots \geq \mu_n$. The algorithm is as follows:

(1) Set weights $w_i^{(0)} = \exp(-\lambda_i^2)$ for all $i = 1, \ldots, m$.

(2) FOR $t = 0$ TO ∞ DO

(3) Define the following subspaces

- $U_1^{(t)} := \text{span}\{e_j : -1 < x_j^{(t)} < 1\}$
- $U_2^{(t)} := \{x \in \mathbb{R}^n \mid \langle x, x^{(t)} \rangle = 0\}$
- $U_3^{(t)} := \{x \in \mathbb{R}^n \mid \langle v_i, x \rangle = 0 \ \forall i \in I^{(t)}\}$. Here $I^{(t)} \subseteq [m]$ are the $|I^{(t)}| = \frac{n}{16}$ indices with maximum weight $w_i^{(t)}$.
- $U_4^{(t)} := \{x \in \mathbb{R}^n \mid \langle v_i, x \rangle = 0 \ \ \forall i \text{ with } \lambda_i \leq 1\}$
- $U_5^{(t)} := \{x \in \mathbb{R}^n \mid \langle x, \sum_{i=1}^m \lambda_i w_i^{(t)} \cdot \exp\left(-\frac{4\delta^2\lambda_i^2}{n}\right) v_i \rangle = 0\}$
- $U_6^{(t)} := \text{span}\{u_j(M^{(t)}) : \frac{15}{16}n \leq j \leq n\}$, for $M^{(t)} := \sum_{i=1}^m w_i^{(t)} \lambda_i^2 v_i v_i^T$.
- $U^{(t)} := U_1^{(t)} \cap \ldots \cap U_6^{(t)}$

(4) Let $z^{(t)}$ be any unit vector in $U^{(t)}$

(5) Choose a maximal $\alpha^{(t)} \in (0, 1]$ so that $x^{(t)} + \delta \cdot y^{(t)} \in [-1, 1]^n$ with $y^{(t)} = \alpha^{(t)} z^{(t)}$.

(6) Update $w_i^{(t+1)} := w_i^{(t)} \cdot \exp(\lambda_i \cdot \delta \cdot \langle v_i, y^{(t)} \rangle) \cdot \exp\left(-\frac{4\delta^2\lambda_i^2}{n}\right)$.

(7) Let $A^{(t)} := \{j \in [n] : -1 < x_j^{(t)} < 1\}$. If $|A^{(t)}| < \frac{n}{2}$, then set $T := t$ and stop.

The intuition is that we maintain weights $w_i^{(t)}$ for each constraint i that increase exponentially with the one-sided discrepancy $\langle v_i, x^{(t)} - x^{(0)} \rangle$. Those weights are discounted in each iteration by a factor that is slightly less than 1 — with a bigger discount for constraints with a larger parameter λ_i. The subspaces $U_1^{(t)}$ and $U_2^{(t)}$ ensure that the length of $x^{(t)}$ is monotonically increasing and fully colored elements remain fully colored.

2.2.1 Bounding the number of iterations

First, note that if the algorithm terminates, then at least half of the variables in $x^{(T)}$ will be either -1 or $+1$. In particular, once a variable is set to ± 1, it is removed from the set $A^{(t)}$ of active variables and the subsequent updates will leave those coordinates invariant.

First we bound the number of iterations. Here we use that the algorithm always makes a step of length δ orthogonal to the current position — except for the steps where it hits the boundary.

Lemma 5. *The algorithm terminates after $T = O(\frac{n}{\delta^2})$ iterations.*

Proof. First, we can analyze the length increase

$$\begin{aligned}\|x^{(t+1)}\|_2^2 &= \|x^{(t)} + \delta \cdot y^{(t)}\|_2^2 \\ &= \|x^{(t)}\|_2^2 + 2\delta \underbrace{\langle x^{(t)}, y^{(t)} \rangle}_{=0} + \delta^2 \|y^{(t)}\|_2^2,\end{aligned}$$

using that $y^{(t)} \in U_2^{(t)}$. Whenever $\alpha^{(t)} = 1$, we have

$$\|x^{(t+1)}\|_2^2 \geq \|x^{(t)}\|_2^2 + \delta^2.$$

It happens that $\alpha^{(t)} < 1$ at most n times, simply because in each such iteration $|A^{(t)}|$ must decrease by at least one. We know that $x^{(T)} \in [-1, 1]^n$.

Now suppose for the sake of contradiction that

$$T > \frac{2n}{\delta^2}.$$

Then

$$\|x^{(T)}\|_2^2 \geq (T - n) \cdot \delta^2 > n,$$

which is impossible. We can hence conclude that the algorithm will terminate in step (7) after at most $\frac{2n}{\delta^2}$ iterations. □

2.2.2 Properties of the subspace $U^{(t)}$

One obvious condition to make the algorithm work is to guarantee that the subspace $U^{(t)}$ satisfies $\dim(U^{(t)}) \geq 1$. In fact, its dimension will even be linear in n.

Lemma 6. *In any iteration t, one has* $\dim(U^{(t)}) \geq \frac{n}{8}$.

Proof. We simply need to account for all linear constraints that define $U^{(t)}$ and we get

$$\begin{aligned}\dim(U^{(t)}) &\geq |A^{(t)}| - |I^{(t)}| - |\{i : \lambda_i \leq 1\}| - \frac{n}{16} - 2 \\ &\geq \frac{n}{2} - \frac{n}{16} - \frac{n}{8} - \frac{n}{16} - 2 \\ &\geq \frac{n}{8},\end{aligned}$$

assuming that $n \geq 16$. □

Another crucial property will be that every vector in $U^{(t)}$ has a bounded *quadratic error term*:

Lemma 7. *For each unit vector $y \in U^{(t)}$ one has*

$$y^T M^{(t)} y \leq \frac{16}{n} \sum_{i=1}^{m} w_i^{(t)} \lambda_i^2.$$

Proof. We have $\mathrm{Tr}[v_i v_i^T] = 1$ since each v_i is a unit vector, hence

$$\begin{aligned} \mathrm{Tr}[M^{(t)}] &= \sum_{i=1}^{m} w_i^{(t)} \lambda_i^2 \mathrm{Tr}[v_i v_i^T] \\ &= \sum_{i=1}^{m} w_i^{(t)} \lambda_i^2. \end{aligned}$$

Because $M^{(t)}$ is positive semidefinite, we know that $\mu_1, \ldots, \mu_n \geq 0$, where $\mu_j := \mu_j(M^{(t)})$ is the jth eigenvalue.

By *Markov's inequality* at most a $\frac{1}{16}$ fraction of eigenvalues can be larger than $\frac{16}{n} \cdot \mathrm{Tr}[M^{(t)}]$. The claim follows as $U_6^{(t)}$ is spanned by the $\frac{15}{16}n$ eigenvectors $v_j(M^{(t)})$ belonging to the smallest eigenvalues, which means

$$\mu_j \leq \frac{16}{n} \mathrm{Tr}[M^{(t)}]$$

for $j = \frac{15}{16}n, \ldots, n$. □

2.2.3 The potential function

So far, we have defined the weights by iterative update steps, but it is not hard to verify that in each iteration t one has the explicit expression

$$w_i^{(t)} = \exp\left(\lambda_i \langle v_i, x^{(t)} - x^{(0)} \rangle - \lambda_i^2 \cdot \left(1 + t \cdot \frac{4\delta^2}{n}\right)\right). \tag{2.1}$$

Inspired by the multiplicative weight update method, we consider the *potential function*

$$\Phi^{(t)} := \sum_{i=1}^{m} w_i^{(t)}$$

that is simply the sum of the individual weights. At the beginning of the algorithm we have

$$\Phi^{(0)} = \sum_{i=1}^{m} w_i^{(0)} = \sum_{i=1}^{m} \exp(-\lambda_i^2/16) \leq \frac{n}{32}$$

using the assumption in Theorem 1. Next, we want to show that the potential function does not increase. Here the choice of the subspaces $U_5^{(t)}$ and $U_6^{(t)}$ will be crucial to control the error.

Lemma 8. *In each iteration t one has $\Phi^{(t+1)} \leq \Phi^{(t)}$.*

Proof. Let us abbreviate $\rho_i := \exp\left(-\frac{4\delta^2\lambda_i^2}{n}\right)$ as the *discount factor* for the ith constant. Note that in particular $0 < \rho_i \leq 1$ and $\rho_i \leq 1 - \frac{2\delta^2\lambda_i^2}{n}$. The change in one step can be analyzed as follows:

$$\begin{aligned}
\Phi^{(t+1)} &= \sum_{i=1}^{m} w_i^{(t+1)} = \sum_{i=1}^{m} w_i^{(t)} \cdot \exp\left(\lambda_i \delta \langle v_i, y^{(t)} \rangle\right) \cdot \rho_i \\
&\overset{(*)}{\leq} \sum_{i=1}^{m} w_i^{(t)} \cdot \left(1 + \lambda_i \delta \langle v_i, y^{(t)} \rangle + \lambda_i^2 \delta^2 \langle v_i, y^{(t)} \rangle^2\right) \cdot \rho_i \\
&= \sum_{i=1}^{m} w_i^{(t)} \cdot \rho_i + \delta \underbrace{\left\langle \sum_{i=1}^{m} \lambda_i w_i^{(t)} \rho_i v_i, y^{(t)} \right\rangle}_{=0 \text{ since } y^{(t)} \in U_5^{(t)}} + \delta^2 \sum_{i=1}^{m} w_i^{(t)} \lambda_i^2 \underbrace{\rho_i}_{\leq 1} \langle v_i, y^{(t)} \rangle^2 \\
&\leq \sum_{i=1}^{m} w_i^{(t)} \cdot \rho_i + \delta^2 \cdot (y^{(t)})^T M^{(t)} y^{(t)} \overset{(**)}{\leq} \sum_{i=1}^{m} w_i^{(t)} \cdot \rho_i + \delta^2 \frac{16}{n} \sum_{i=1}^{m} w_i^{(t)} \lambda_i^2 \\
&\overset{(***)}{\leq} \sum_{i=1}^{m} w_i^{(t)} = \Phi^{(t)}.
\end{aligned}$$

In $(*)$, we use the inequality $e^x \leq 1 + x + x^2$ for $|x| \leq 1$ together with the fact that $\lambda_i \delta |\langle v_i, y^{(t)} \rangle| \leq \lambda_i \delta \leq 1$. In $(**)$ we bound $(y^{(t)})^T M^{(t)} y^{(t)}$ using Lemma 7. In $(***)$ we finally use the fact that $\rho_i + \frac{16}{n}\delta^2 \leq 1$. □

Typically in the multiplicative weight update method one can only use the fact that $\max_{i \in [m]} w_i^{(t)} \leq \Phi^{(t)}$ which would lead to the loss of an additional $\sqrt{\log n}$ factor. The trick in our approach is that there is always a *linear* number of weights of order $\max_{i \in [m]} w_i^{(t)}$ since the updates are always chosen orthogonal to the $\frac{n}{16}$ constraints with highest weight.

Lemma 9. *At the end of the algorithm,* $\max\{w_i^{(T)} : i \in [m]\} \leq 2$.

Proof. Suppose, for contradiction, that $w_i^{(T)} > 2$ for some i. Let t^* be the last iteration when i was not among the $\frac{n}{16}$ constraints with highest weight. Then

$$2 < w_i^{(T)} = w_i^{(t^*+1)} = w_i^{(t^*)} \cdot \underbrace{\exp(\lambda_i \cdot \delta \cdot \langle v_i, y^{(t)} \rangle)}_{\leq e} \cdot \underbrace{\rho_i}_{\leq 1} \leq w_i^{(t^*)} \cdot e,$$

and hence, $w_i^{(t^*)} > \frac{2}{e}$. This would imply that $\Phi^{(t^*)} \geq \frac{n}{16} \cdot \frac{2}{e} > \frac{n}{32}$, contradicting Lemma 8. □

Lemma 10. *If* $w_i^{(T)} \leq 2$, *then* $\langle v_i, x^{(T)} - x^{(0)} \rangle \leq 11\lambda_i$.

Proof. First note that the algorithm always walks orthogonal to all constraint vectors v_i if $\lambda_i \leq 1$ and in this case $\langle v_i, x^{(T)} - x^{(0)} \rangle = 0$. Now suppose that $\lambda_i > 1$. We know that

$$w_i^{(T)} \overset{(2.1)}{=} \exp\left(\lambda_i \cdot \langle v_i, x^{(T)} - x^{(0)} \rangle - \lambda_i^2 \cdot \left(1 + 4 \cdot T \cdot \frac{\delta^2}{n}\right)\right) \leq 2.$$

Taking logarithms on both sides and dividing by λ_i then gives

$$\langle v_i, x^{(T)} - x^{(0)} \rangle \leq \underbrace{\frac{\log(2)}{\lambda_i}}_{\leq 2} + \lambda_i \Big(1 + 4 \underbrace{T \frac{\delta^2}{n}}_{\leq 2}\Big) \leq 11\lambda_i.$$

This lemma concludes the proof of Theorem 1. □

2.2.4 Application to set coloring

Now we come to the main application of the partial coloring argument from Theorem 1, which is to color set systems:

Lemma 11. *Given a set system* $S_1, \ldots, S_m \subseteq [n]$, *we can find a coloring* $x \in \{-1, 1\}^n$ *with* $|\sum_{j \in S_i} x_j| \leq O\left(\sqrt{n \log \frac{2m}{n}}\right)$ *for every* i *deterministically in time* $O\big(n^3 m \log(\frac{2m}{n})\big)$.

Proof. For a fractional vector x, let us abbreviate

$$\operatorname{disc}(S, x) := \left|\sum_{j \in S} x_j\right|$$

as the discrepancy with respect to set S. Set $x^{(0)} := 0$. For $s = 1, \ldots, \log_2(n)$ many phases we do the following. Let

$$A^{(s)} := \{i \in [n] : -1 < x_i^{(s-1)} < 1\}$$

be the not yet fully colored elements. Define a vector

$$v_i := \frac{1}{\sqrt{|A^{(s)}|}} 1_{S_i \cap A^{(s)}}$$

of length $\|v_i\|_2 \leq 1$ with parameters

$$\lambda_i := C\sqrt{\log\left(\frac{2m}{|A^{(s)}|}\right)}.$$

Then apply Theorem 1 to find $x^{(s)} \in [-1, 1]$ with

$$\operatorname{disc}(S_i, x^{(s)} - x^{(s-1)}) \leq O\left(\sqrt{|A^{(s)}| \log\left(\frac{2m}{|A^s|}\right)}\right)$$

such that $x_i^{(s)} = x_i^{(s-1)}$ for $i \notin A^{(s)}$. Since each time at least half of the elements get fully colored we have $|A^{(s)}| \leq 2^{-(s-1)}n$ for all s. Then

$$x := x^{(\log_2 n)} \in \{-1, 1\}^n$$

and

$$\operatorname{disc}(S_i, x) \leq \sum_{s \geq 1} O\left(\sqrt{2^{-(s-1)} n \log\left(\frac{2m}{2^{-(s-1)}n)}\right)}\right) \leq O\Big(\sqrt{n \log(\tfrac{2m}{n})}\Big)$$

using that this convergent sequence is dominated by the first term.

In each application of Theorem 1 one has $\delta \geq \Omega(1/\sqrt{\log(\frac{2m}{n})})$. Thus phase s runs for

$$O(2^{-(s-1)}n/\delta^2) = O\left(2^{-(s-1)} n \log\left(\frac{2m}{n}\right)\right)$$

iterations, each of which takes $O((2^{-(s-1)}n)^2 m)$ time, for a total runtime of

$$O\left(\left(2^{-(s-1)}n\right)^3 m \log\left(\frac{2m}{n}\right)\right)$$

in phase s. Summing the geometric series for $s = 1, \ldots, \log_2 n$ results in a total running time of $O(n^3 m \log(\frac{2m}{n}))$. □

By setting $m = n$ in Lemma 11, we obtain Corollary 2.

2.3 Matrix balancing

In this section we prove Theorem 3. We begin with some preliminaries.

For matrices $A, B \in \mathbb{R}^{n\times n}$, let

$$A \bullet B := \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij} \cdot B_{ij}$$

be the *Frobenius inner product.*

Recall that any symmetric matrix $A \in \mathbb{R}^{n\times n}$ can be written as

$$A = \sum_{j=1}^{n} \mu_j u_j u_j^T,$$

where μ_j is the eigenvalue corresponding to eigenvector u_j.

The *trace* of A is

$$\mathrm{Tr}[A] = \sum_{i=1}^{n} A_{ii} = \sum_{j=1}^{n} \mu_j$$

and for symmetric matrices A, B one has $\mathrm{Tr}[AB] = A \bullet B$.

If A has only nonnegative eigenvalues, we say that A is *positive semidefinite* and write $A \succeq 0$. Recall that $A \succeq 0$ if and only if $y^T A y \geq 0$ for all $y \in \mathbb{R}^n$.

For a symmetric matrix A, we denote $\mu_{\max} := \max\{\mu_j : j = 1, \ldots, n\}$ as the largest Eigenvalue and $\|A\|_{\mathrm{op}} := \max\{|u_j| : j = 1, \ldots, n\}$ as the largest singular value.

Note that if $A \succeq 0$, then $|A \bullet B| \leq \mathrm{Tr}[A] \cdot \|B\|_{\mathrm{op}}$. If $A, B \succeq 0$, then $A \bullet B \geq 0$. Finally, note that for any symmetric matrix A one has $A^2 := AA \succeq 0$.

From the eigendecomposition $A = \sum_{j=1}^{n} \mu_j u_j u_j^T$, one can easily show that the maximum singular value also satisfies

$$\|A\|_{\mathrm{op}} = \max\{\|Ay\|_2 : \|y\|_2 = 1\}$$

and

$$\|A\|_{\mathrm{op}} = \max\{|y^T A y| : \|y\|_2 = 1\}.$$

For any function $f : \mathbb{R} \to \mathbb{R}$ we define

$$f(A) := \sum_{j=1}^{n} f(\mu_j) u_j u_j^T$$

to be the symmetric matrix that is obtained by applying f to all Eigenvalues. In particular, we will be interested in the *matrix exponential*

$$\exp(A) := \sum_{j=1}^{n} e^{\mu_j} u_j u_j^T.$$

For any symmetric matrices $A, B \in \mathbb{R}^{n \times n}$, the *Golden-Thompson inequality* says that

$$\mathrm{Tr}[\exp(A+B)] \leq \mathrm{Tr}[\exp(A)\exp(B)].$$

(It is not hard to see that for diagonal matrices one has equality.) We refer to the textbook of Bhatia [11] for more details.

Theorem 12. *Let $A_1, \ldots, A_n \in \mathbb{R}^{m \times m}$ be q-block diagonal matrices with $\|A_i\|_{op} \leq 1$ for $i = 1, \ldots, m$ and let $x^{(0)} \in [-1,1]^n$ be a starting point. Then there is a deterministic algorithm that finds an $x \in [-1,1]^n$ with*

$$\Big\| \sum_{i=1}^{n} (x_i - x_i^{(0)}) \cdot A_i \Big\|_{op} \leq O\Big(\sqrt{n \log\Big(\frac{2qm}{n}\Big)}\Big)$$

in time $O(n^5 + n^4 m^3)$. Moreover, at least $\frac{n}{2}$ coordinates of x will be in $\{-1, 1\}$.

Our algorithm computes a sequence of iterates $x^{(0)}, \ldots, x^{(T)}$ such that $x^{(T)}$ is the desired vector x with half of the coordinates being integral. In our algorithm the step size is $\delta = \frac{1}{\sqrt{n}}$ and we use a parameter $\varepsilon = \frac{1}{\sqrt{n}}$ to control the scaling of the following potential function:

$$\Phi^{(t)} := \mathrm{Tr}\Big[\exp\Big(\varepsilon \sum_{i=1}^{n} (x_i^{(t)} - x_i^{(0)}) \cdot A_i \Big) \Big].$$

Suppose $B_{i,k} \in \mathbb{R}^{q \times q}$ are symmetric matrices so that $A_i = \mathrm{diag}(B_{i,1}, \ldots, B_{i,m/q})$. Then we can decompose the weight function as

$$\Phi^{(t)} = \sum_{k=1}^{m/q} \Phi_j^{(t)}$$

with

$$\Phi_k^{(t)} := \mathrm{Tr}\Big[\exp\Big(\varepsilon \sum_{i=1}^{n} (x_i^{(t)} - x_i^{(0)}) B_{i,k}\Big)\Big].$$

In other words, the potential function is simply the sum of the potential function applied to each individual block.

The algorithm is as follows:

(1) FOR $t = 0$ TO ∞ DO

(2) Define weight matrix $W^{(t)} := \exp(\varepsilon \sum_{i=1}^{n} (x_i^{(t)} - x_i^{(0)}) A_i)$

(3) Define the following subspaces

- $U_1^{(t)} := \mathrm{span}\{e_j : -1 < x_j^{(t)} < 1\}$
- $U_2^{(t)} := \{x \in \mathbb{R}^n \mid \langle x, x^{(t)} \rangle = 0\}$
- $U_3^{(t)} := \{x \in \mathbb{R}^n \mid \sum_{i=1}^{n} x_i B_{i,k} = 0 \ \forall k \in I^{(t)}\}$. Here $I^{(t)} \subseteq [m]$ are the $|I^{(t)}| = \frac{1}{16} \cdot \frac{n}{q^2}$ indices k with maximum weight $\Phi_k^{(t)}$.
- $U_4^{(t)} := \{x \in \mathbb{R}^n \mid \sum_{i=1}^{n} x_i \cdot (W^{(t)} \bullet A_i) = 0\}$
- $U_5^{(t)}$ is the subspace defined in Lemma 14, with $k = 16$.
- $U^{(t)} := U_1^{(t)} \cap \ldots \cap U_5^{(t)}$

(4) Let $z^{(t)}$ be any unit vector in $U^{(t)}$.

(5) Choose a maximal $\alpha^{(t)} \in (0, 1]$ so that $x^{(t)} + \delta \cdot y^{(t)} \in [-1, 1]^n$, where $y^{(t)} = \alpha^{(t)} z^{(t)}$.

(6) Let $A^{(t)} := \{j \in [n] : -1 < x_j^{(t)} < 1\}$. If $|A^{(t)}| < \frac{n}{2}$, then set $T := t$ and stop.

The analysis of our algorithm follows a sequence of lemmas. First, by exactly the same arguments as in Lemma 5 we know that the algorithm terminates after $T \leq \frac{2n}{\delta^2}$ iterations. Each iteration can be completed in time $O(n^2 m^3 + n^3)$ (c.f. Lemma 14).

Lemma 13. *In each iteration t one has* $\dim(U^{(t)}) \geq \frac{n}{4}$.

Proof. We simply need to account for all linear constraints that define $U^{(t)}$ and we get

$$\begin{aligned}\dim(U^{(t)}) &\geq \underbrace{|A^{(t)}|}_{U_1^{(t)}} - \underbrace{|I^{(t)}|}_{U_3^{(t)}} - \underbrace{\frac{n}{16}}_{U_5^{(t)}} - \underbrace{2}_{U_2^{(t)},U_4^{(t)}} \\ &\geq \frac{n}{2} - \frac{n}{16q^2} \cdot q^2 - \frac{n}{16} - 2 \\ &\geq \frac{n}{4},\end{aligned}$$

assuming that $n \geq 16$. □

To analyze the behavior of the potential function, we first prove the existence of a suitable subspace $U_5^{(t)}$ that will bound the quadratic error term.

Lemma 14. *Let $W \in \mathbb{R}^{m\times m}$ be a symmetric positive semidefinite matrix, let $A_1, \ldots, A_n \in \mathbb{R}^{m\times m}$ be symmetric matrices with $\|A_i\|_{op} \leq 1$ and let $k > 0$ be a parameter. Then in time $O(n^2m^3 + n^3)$ one can compute a subspace $U \subseteq \mathbb{R}^n$ of dimension $\dim(U) \geq (1 - \frac{1}{k})n$ so that*

$$W \bullet \Big(\sum_{i=1}^{n} y_i A_i\Big)^2 \leq k \cdot Tr[W] \quad \forall y \in U \text{ with } \|y\|_2 = 1. \tag{2.2}$$

Proof. Let $M \in \mathbb{R}^{n\times n}$ be the matrix with entries $M_{ij} := W \bullet A_iA_j$ for all $i, j \in [n]$. First note that the matrix M is symmetric[3], because

$$M_{ij} = W \bullet A_iA_j = W^T \bullet (A_iA_j)^T = W \bullet A_jA_i = M_{ji}.$$

Now, for any $y \in \mathbb{R}^n$, $\left(\sum_{i=1}^n y_iA_i\right)^2$ is symmetric and positive semidefinite and hence

$$y^TMy = \sum_{i,j=1}^{n} y_iy_j(W \bullet A_iA_j) = W \bullet \Big(\sum_{i=1}^{n} y_iA_i\Big)^2 \geq 0,$$

proving that M is positive semidefinite.

Consider the eigendecomposition

$$M = \sum_{i=1}^{n} \mu_i \, u_iu_i^T$$

[3]One has to be careful as the product A_iA_j is in general *not* symmetric, even if A_i and A_j are symmetric.

where $\mu_i \geq 0$. Define the subspace

$$U := \mathrm{span}\{u_i \colon \mu_i < k\mathrm{Tr}[W]\}.$$

The desired inequality (2.2) follows immediately from the definition of U. All that remains is to verify that $\dim(U) \geq (1 - \frac{1}{k})n$. Since $\mu_i \geq 0$ we may apply *Markov's inequality* to deduce that

$$\#\{i \colon \mu_i \geq k \cdot \mathrm{Tr}[W]\} \leq \frac{\mathrm{Tr}[M]}{k\mathrm{Tr}[W]} \leq \frac{n}{k},$$

where in the second inequality we have used the bound

$$\mathrm{Tr}[M] = \sum_{i=1}^{n} M_{ii} = \sum_{i=1}^{n} W \bullet A_i^2 \leq n \cdot \mathrm{Tr}[W].$$

Hence

$$\dim(U) \geq n - \#\{i \colon \mu_i \geq k\mathrm{Tr}[W]\} \geq \left(1 - \frac{1}{k}\right) n,$$

as desired.

Finally, to bound the running time, we observe that computing M takes time $O(n^2m^3)$ and the eigendecomposition of M can be computed in time $O(n^3)$. □

Again, we bound the increase in the potential function:

Lemma 15. *In each iteration t, one has $\Phi^{(t+1)} \leq (1 + 16\varepsilon^2\delta^2) \cdot \Phi^{(t)}$.*

Proof. We estimate that

$$\begin{aligned}
\Phi^{(t+1)} &= \mathrm{Tr}\Big[\exp\Big(\varepsilon\sum_{i=1}^n (x_i^{(t+1)} - x_i^{(0)})A_i\Big)\Big] \\
&= \mathrm{Tr}\Big[\exp\Big(\varepsilon\sum_{i=1}^n (x_i^{(t)} - x_i^{(0)})A_i + \varepsilon\delta\sum_{i=1}^n y_i^{(t)}\Big)\Big] \\
&\overset{(*)}{\leq} \mathrm{Tr}\Big[\underbrace{\exp\Big(\varepsilon\sum_{i=1}^n (x_i^{(t)} - x_i^{(0)})\cdot A_i\Big)}_{=W^{(t)}}\exp\Big(\varepsilon\delta\sum_{i=1}^n y_i^{(t)}A_i\Big)\Big] \\
&= W^{(t)} \bullet \exp\Big(\varepsilon\delta\sum_{i=1}^n y_i^{(t)}A_i\Big) \\
&\overset{(**)}{\leq} W^{(t)} \bullet \Big(I + \varepsilon\delta\sum_{i=1}^n y_i^{(t)}A_i + \varepsilon^2\delta^2\Big(\sum_{i=1}^n y_i^{(t)}A_i\Big)^2\Big) \\
&= \underbrace{W^{(t)} \bullet I}_{=\Phi^{(t)}} + \varepsilon\delta\underbrace{\Big(W^{(t)} \bullet \sum_{i=1}^n y_i^{(t)}A_i\Big)}_{=0 \text{ since } y^{(t)}\in U_4^{(t)}} + \varepsilon^2\delta^2\underbrace{\Big(W^{(t)} \bullet \Big(\sum_{i=1}^n y_i^{(t)}A_i\Big)^2\Big)}_{\leq 16\cdot\mathrm{Tr}[W^{(t)}]} \\
&\overset{(***)}{\leq} \Phi^{(t)}\cdot(1 + 16\varepsilon^2\delta^2).
\end{aligned}$$

In $(*)$ we use the Golden-Thompson inequality. In $(**)$ we use that $\exp(X) \preceq I + X + X^2$ for any symmetric matrix X with $\|X\|_{\mathrm{op}} \leq 1$ together with the triangle inequality

$$\left\|\varepsilon\delta\sum_{i=1}^n y_i^{(t)}A_i\right\|_{op} \leq \varepsilon\delta\sum_{i=1}^n |y_i^{(t)}|\cdot\|A_i\|_{op} \leq \varepsilon\delta n = 1.$$

In $(***)$ we use Lemma 14 and the fact that $y^{(t)} \in U_5^{(t)}$. □

This gives us a bound on the potential function at the end of the algorithm.

Lemma 16. *At the end of the algorithm,* $\Phi^{(T)} \leq m\cdot\exp(32\varepsilon^2 n)$.

Proof. Since

$$\Phi^{(0)} = \mathrm{Tr}[\exp(0)] = \mathrm{Tr}[I] = m,$$

we get that

$$\Phi^{(T)} \leq m\cdot(1 + 16\varepsilon^2\delta^2)^T \leq m\cdot\exp(32\varepsilon^2 n),$$

using the fact that $T \leq \frac{2n}{\delta^2}$. □

Lemma 17. *We have* $\mu_{\max}\left(\sum_{i=1}^{n}(x_i^{(T)} - x_i^{(0)}) \cdot A_i\right) = O\left(\sqrt{n\log(\frac{2qm}{n})}\right)$.

Proof. Let

$$\mu_{\max} := \mu_{\max}\left(\sum_{i=1}^{n}(x_i^{(T)} - x_i^{(0)}) \cdot A_i\right).$$

Suppose the eigenspace corresponding to $\mu_{\max}$ lies in block k, for some $k \in \{1, \ldots, m/q\}$. Let t^* be the last iteration when k was not among the $n/(16q^2)$ indices with maximum weight. We then have

$$\begin{aligned}
\Phi_k^{(T)} &= \Phi_k^{(t^*+1)} \\
&= \mathrm{Tr}\Big[\exp\Big(\varepsilon\sum_{i=1}^{n}(x_i^{(t^*)} + \delta y_i^{(t^*)} - x_i^{(0)})B_{i,k}\Big)\Big] \\
&\overset{(*)}{\leq} \mathrm{Tr}\Big[\exp\Big(\varepsilon\sum_{i=1}^{n}(x_i^{(t^*)} - x_i^{(0)})B_{i,k}\Big)\underbrace{\exp\Big(-\varepsilon\delta\sum_{i=1}^{n}y_i^{(t^*)}B_{i,k}\Big)}_{\preceq e\mathbf{I} \quad (**)}\Big] \\
&\leq e\cdot\Phi_k^{(t^*)},
\end{aligned}$$

where in $(*)$, we use the Golden-Thompson inequality. In $(**)$ we have used the bounds $\|B_{i,k}\|_{op} \leq \|A_i\|_{op} \leq 1$ and $|y_i^{(t^*)}| \leq 1$ together with the triangle inequality to deduce that

$$\left\|\varepsilon\delta\sum_{i=1}^{n}y_i^{(t^*)}B_{i,k}\right\|_{op} \leq 1.$$

Hence

$$e^{\varepsilon\mu_{\max}} \leq \Phi_k^{(T)} \leq e\cdot\Phi_k^{(t^*)} \leq e\cdot\frac{16q^2}{n}\cdot\Phi^{(T)} \leq \frac{16eq^2}{n}\cdot m\exp(32\varepsilon^2 n).$$

Then taking logarithms and dividing by ε gives

$$\mu_{\max} \leq \frac{1}{\varepsilon}\cdot\log\left(\frac{16eq^2m}{n}\right) + 32\varepsilon n = O\Big(\sqrt{n\log\left(\frac{qm}{n}\right)}\Big),$$

where in the final inequality we have used that $\varepsilon = \sqrt{\frac{\log(qm/n)}{n}}$. □

These lemmas put together give us Theorem 12: an algorithm that yields a partial coloring with the claimed properties. We run the algorithm in phases to obtain Theorem 3, by boosting the partial coloring to a full coloring using a similar technique as in Lemma 11.

Proof of Theorem 3. Set $x^{(0)} := 0$. For $s = 1, \ldots, \log_2(n)$ many phases we do the following. Let

$$J^{(s)} := \{i \in [n] : -1 < x_i^{(s-1)} < 1\}$$

be the not yet fully colored elements. Apply Theorem 12 to find $x^{(s)} \in [-1, 1]^n$ with

$$\left\| \sum_{i \in J^{(s)}} (x_i^{(s)} - x_i^{(s-1)}) \cdot A_i \right\|_{op} = O\left(\sqrt{|J^{(s)}| \log \frac{2qm}{|J^s|}} \right),$$

and such that $x_i^{(s)} = x_i^{(s-1)}$ for all $i \notin J^{(s)}$. Since each time at least half of the elements get fully colored we have $|J^{(s)}| \leq 2^{-(s-1)}n$ for all s. Then $x := x^{(\log_2 n)} \in \{-1, 1\}^n$ and

$$\left\| \sum_{i=1}^{n} x_i \cdot A_i \right\|_{op} = \sum_{s \geq 1} O\Big(\sqrt{2^{-(s-1)} n \log \left(\frac{2qm}{2^{-(s-1)}n} \right)} \Big) = O\Big(\sqrt{n \log(\tfrac{2qm}{n})} \Big),$$

using that the sum of a subgeometric sequence is dominated by its first term. Phase s has a running time of $O((2^{-(s-1)}n)^5 + (2^{-(s-1)}n)^4 m^3)$ and summing this geometric series over $s = 1, \ldots, \log_2 n$ yields a total runtime of $O(n^5 + n^4 m^3)$. □

2.4 Discrepancy minimization for matrices with bounded column length

In this section we prove Theorem 4. Fix a matrix $A \in \mathbb{R}^{m\times n}$ with $\|A^j\|_2 \leq 1$ for each column $j = 1, \ldots, n$. Recently Bansal, Dadush and Garg [7] gave the first polynomial time algorithm to find a coloring $x \in \{-1, 1\}^n$ with $\|Ax\|_\infty \leq O(\sqrt{\log n})$. Their method is based on a random walk, where the random updates in each iteration are chosen using a semidefinite program that has to be re-solved each time.

We show that instead a deterministic walk can be used, guided by a suitable exponential potential function. The update directions will be chosen from the intersection of subspaces satisfying certain constraints; no SDP has to be solved in our method. We should also mention that the more general non-constructive result of Banaszczyk [5] even guarantees signs x so that $Ax \in 5 \cdot K$, where K is any convex body with Gaussian measure at least $1/2$.

In this section, let $C > 0$ be a sufficiently large constant. For a row i with $\|A_i\|_2^2 \leq \frac{1}{n}$, any coloring x will satisfy

$$|\langle A_i, x\rangle| \leq \|A_i\|_2 \cdot \|x\|_2 \leq 1$$

and we can safely remove such a row. From now on we can assume that

$$\|A_i\|_2^2 \geq \frac{1}{n}$$

and hence $m \leq n^2$.

Note that it also suffices to find an $x \in \{-1, 1\}^n$ satisfying the *one-sided* error

$$\langle A_i, x\rangle \leq O(\sqrt{\log n})$$

as one can simply stack $\frac{1}{\sqrt{2}}A$ and $-\frac{1}{\sqrt{2}}A$ together.

Next, replace each row A_i with two rows: one row is the *light* row containing all entries of size

$$|A_{ij}| \leq \frac{1}{C^2\sqrt{\log n}}$$

and the other row is the *heavy* row whose only nonzero entries have size

$$|A_{ij}| > \frac{1}{C^2\sqrt{\log n}}.$$

After this modification, we abbreviate the indices as

$$I_{\text{light}} := \left\{ i \in [m] : \|A_i\|_\infty \leq \frac{1}{C^2\sqrt{\log n}} \right\}$$

and

$$I_{\text{heavy}} := \left\{ i \in [m] : \|A_i\|_\infty > \frac{1}{C^2\sqrt{\log n}} \right\}.$$

As in the previous settings, our algorithm will compute a sequence

$$x^{(0)}, \ldots, x^{(T)} \in [-1, 1]^n,$$

starting at $x^{(0)} = 0$ so that the final point $x^{(T)}$ has coordinates only in $\{-1, 1\}$.

For the point $x^{(t)} \in [-1, 1]^n$ and some parameters $\alpha, \beta > 0$ that we specify later, we define a *potential function*

$$\Phi^{(t)} := \sum_{i \in I_{\text{light}}} w_i^{(t)}$$

with

$$w_i^{(t)} := \exp\left(\alpha \left\langle A_i, x^{(t)} \right\rangle + \beta \min\left\{ C, \sum_{j=1}^{n} (1 - (x_j^{(t)})^2) \cdot A_{ij}^2 \right\} \right).$$

Here the quantity

$$L(i, x) := \sum_{j=1}^{n} (1 - x_j^2) \cdot A_{ij}^2$$

can be interpreted as the *effective length* of row i with $L(i, 0) = \|A_i\|_2^2$ and $L(i, x) = 0$, if $x \in \{-1, 1\}^n$.

The intuition behind the algorithm is as follows: at the beginning one has $x^{(0)} = 0$ and the whole weight of the potential function comes from the β-term. Then in the course of the algorithm the weight is transferred from the β-term to the α-term until all elements are colored and the effective length of all constraints is 0. In fact, if $\beta \geq \Omega(\alpha^2)$, we show that the potential function is nonincreasing.

To keep the notation readable, for vectors $x, y \in \mathbb{R}^n$ we write $(x \circ y) \in \mathbb{R}^n$ for the vector with components $(x \circ y)_i := x_i \cdot y_i$ and $x^{\circ 2} := x \circ x$. Moreover, $x^{\otimes 2} := xx^T$ is the *tensor product*.

As before, we find an update vector in each iteration so that the potential function does not increase, by choosing it from the intersection of certain subspaces. We postpone some linear algebra arguments till Section 2.4.2.

We use the following algorithm:

(1) Set $x^{(0)} := 0$ and $A^{(0)} := [n]$.

(2) FOR $t = 0$ TO T DO

(3) Let $I^{(t)} := \{i \in I_{\text{light}} \mid L(i, x^{(t)}) < C\} \cup \{i \in I_{\text{heavy}} \mid \sum_{j \in A^{(t)}} A_{ij}^2 < C\}$. Define the subspaces

- $U_0^{(t)} := \text{span}\{e_j \mid j \in A^{(t)}\} \subseteq \mathbb{R}^n$
- $U_1^{(t)} := \{x \in U_0^{(t)} \mid \langle x, x^{(t)} \rangle = 0\}$
- $U_2^{(t)} := \{x \in U_0^{(t)} \mid \langle x, A_i \rangle = 0 \; \forall i \notin I^{(t)}\}$
- $U_3^{(3)} := \{x \in U_0^{(t)} \mid \left\langle \sum_{i \in I^{(t)}} w_i^{(t)} A_i, x \right\rangle = 0\}$
- $U_4^{(t)} := \{x \in U_0^{(t)} \mid \left\langle \sum_{i \in I^{(t)}} w_i^{(t)} \cdot (A_i^{\circ 2} \circ x^{(t)}), x \right\rangle = 0\}$
- $U_5^{(t)} \subseteq U_0^{(t)}$ with $\sum_{i \in I^{(t)}} w_i^{(t)} \langle A_i, x \rangle^2 \leq \frac{\beta}{16\alpha^2} \sum_{j=1}^n x_j^2 A_{ij}^2$ for all $x \in U_5^{(t)}$ and $\dim(U_5^{(t)}) \geq \frac{15}{16}|A^{(t)}|$ (see Sec. 2.4.2)
- $U_6^{(t)} \subseteq U_0^{(t)}$ with $\dim(U_6^{(t)}) \geq \frac{15}{16}|A^{(t)}|$ and $\sum_{i \in I^{(t)}} w_i^{(t)} \left\langle A_i^{\circ 2} \circ x^{(t)}, x \right\rangle^2 \leq \frac{1}{8\beta} \sum_{j=1}^n x_j^2 A_{ij}^2$ for all $x \in U_6^{(t)}$ (see Sec. 2.4.2)
- $U^{(t)} := U_1^{(t)} \cap \ldots \cap U_6^{(t)}$

(4) Let $z^{(t)}$ be any unit vector in $U^{(t)}$.

(5) Choose a maximal $\alpha^{(t)} \in (0, 1]$ so that $x^{(t+1)} := x^{(t)} + \delta \cdot y^{(t)} \in [-1, 1]^n$ with $y^{(t)} = \alpha^{(t)} z^{(t)}$.

(6) Let $A^{(t)} := \{j \in [n] : -1 < x_j^{(t)} < 1\}$. If $|A^{(t)}| \leq C$, then set $T := t$ and stop.

Technically speaking, the final point $x^{(T)}$ still has a constant number of entries not in $\{-1, 1\}$ — these entries can be rounded arbitrarily. The first step is to guarantee that the subspace $U^{(t)}$ is indeed non-empty in each iteration:

Lemma 18. *In each iteration t, we have $\dim(U^{(t)}) \geq \frac{1}{2}|A^{(t)}|$, if C is chosen large enough.*

Proof. Observe that for $i \in I_{\text{light}} \setminus I^{(t)}$,

$$\begin{aligned}\sum_{j \in A^{(t)}} A_{ij}^2 &\geq \sum_{j \in A^{(t)}} (1 - \big(x_j^{(t)}\big)^2) A_{ij}^2 \\ &= \sum_{j=1}^{n} (1 - \big(x_j^{(t)}\big)^2) A_{ij}^2 \\ &= L(i, x^{(t)}) \\ &\geq C,\end{aligned}$$

and hence $\sum_{j \in A^{(t)}} A_{ij}^2 \geq C$ holds for all $i \notin I^{(t)}$. Now, since the l^2-norm of each column A^j is at most 1, we have

$$\begin{aligned}|A^{(t)}| &\geq \sum_{j \in A^{(t)}} \underbrace{\sum_{i \in [m]} A_{ij}^2}_{\leq 1} = \sum_{i \in [m]} \sum_{j \in A^{(t)}} A_{ij}^2 \\ &\geq \sum_{i \notin I^{(t)}} \underbrace{\sum_{j \in A^{(t)}} A_{ij}^2}_{\geq C} \geq C(m - |I^{(t)}|).\end{aligned}$$

Hence, $\operatorname{codim}(U_2^{(t)}) \leq |A^{(t)}|/C$. We can hence bound

$$\dim(U^{(t)}) \geq \underbrace{|A^{(t)}|}_{U_0^{(t)}} - \underbrace{\frac{|A^{(t)}|}{C}}_{U_2^{(t)}} - \underbrace{\frac{|A^{(t)}|}{16}}_{U_5^{(t)}} - \underbrace{\frac{|A^{(t)}|}{16}}_{U_6^{(t)}} - \underbrace{(1+1+1)}_{U_1^{(t)}, U_3^{(t)}, U_4^{(t)}} \geq \frac{|A^{(t)}|}{2},$$

if C is chosen large enough. □

As before, one always has $\|x^{(t+1)}\|_2^2 \geq \|x^{(t)}\|_2^2$ and in each but at most n iterations one has $\|x^{(t+1)}\|_2^2 = \|x^{(t)}\|_2^2 + \delta^2$. Then the algorithm terminates after $T \leq n + \frac{n}{\delta^2} \leq \frac{2n}{\delta^2}$ iterations, given that $0 < \delta \leq 1$.

The main part of the analysis lies in guaranteeing that the potential function is nonincreasing.

Lemma 19. *Suppose that $\beta \geq C \cdot \alpha^2$ where $C > 0$ is a large enough constant with $0 < \delta \leq \frac{1}{36\sqrt{\beta}}$ and $\|A_i\|_\infty \leq \frac{1}{C\sqrt{\beta}}$ for $i \in I_{light}$. Then in each iteration t we have $\Phi^{(t+1)} \leq \Phi^{(t)}$.*

Proof. Note that $w_i^{(t+1)} \leq w_i^{(t)}$ for any light index with $L(i, x^{(t)}) \geq C$. In fact, one can only have strict inequality if $L(i, x^{(t)}) \geq C > L(i, x^{(t+1)})$.

Hence we only need to prove that

$$\sum_{i \in I^{(t)} \cap I_{\text{light}}} w_i^{(t+1)} \leq \sum_{i \in I^{(t)} \cap I_{\text{light}}} w_i^{(t)}.$$

For ease of notation, we drop the index t and also write $x' = x + \delta y$ instead of $x^{(t+1)} = x^{(t)} + \delta y^{(t)}$, and I instead of $I^{(t)} \cap I_{\text{light}}$.

We estimate that

$$\begin{aligned}
\sum_{i \in I} w_i^{(t+1)} &= \sum_{i \in I} \exp\Big(\alpha \langle A_i, x + \delta y\rangle + \beta \sum_{j=1}^{n} (1 - (x_j + \delta y_j)^2) \cdot A_{ij}^2\Big) && (2.3)\\
&= \sum_{i \in I} \underbrace{\exp\Big(\alpha \langle A_i, x\rangle + \beta \sum_{j=1}^{n} (1 - x_j^2) \cdot A_{ij}^2\Big)}_{=w_i} && (2.4)\\
&\quad \cdot \exp\Big(\alpha\delta \langle A_i, y\rangle - 2\beta\delta \big\langle A_i^{\circ 2} \circ x, y\big\rangle - \beta\delta^2 \sum_{j=1}^{n} y_j^2 A_{ij}^2\Big)
\end{aligned}$$

Now we bound the second exponential term using the inequality $e^{x_1+x_2+x_3} \leq 1 + x_1 + x_2 + x_3 + 9x_1^2 + 9x_2^2 + 9x_3^2$ for $\max\{|x_1|, |x_2|, |x_3|\} \leq 1$. We obtain

$$\begin{aligned}
(2.3) \;\leq\; & \sum_{i \in I} w_i + \Big[\alpha\delta \underbrace{\sum_{i \in I} w_i \cdot \langle A_i, y\rangle}_{=0 \text{ as } y \in U_3^{(t)}} + 9\alpha^2\delta^2 \sum_{i \in I} w_i \cdot \langle A_i, y\rangle^2\Big] \\
& + \Big[-2\beta\delta \underbrace{\sum_{i \in I} w_i \cdot \big\langle A_i^{\circ 2} \circ x, y\big\rangle}_{=0 \text{ as } y \in U_4^{(t)}} + 9 \cdot 4\beta^2\delta^2 \sum_{i \in I} w_i \cdot \big\langle A_i^{\circ 2} \circ x, y\big\rangle^2\Big] \\
& + \underbrace{\Big[-\beta\delta^2 \sum_{i \in I} w_i \sum_{j=1}^{n} y_j^2 A_{ij}^2 + 9\beta^2\delta^4 \sum_{i \in I} w_i \cdot \Big(\sum_{j=1}^{n} y_j^2 A_{ij}^2\Big)^2\Big]}_{\leq -\frac{\beta}{2}\delta^2 \sum_{i \in I} w_i \sum_{j=1}^{n} y_j^2 A_{ij}^2}
\end{aligned}$$

Now, we use the fact that $9\beta\delta^2 \sum_{j=1}^n y_j^2 A_{ij}^2 \le 9\beta\delta^2 \le \frac{1}{2}$ to get

$$\begin{aligned}(2.3) \quad &\le \sum_{i\in I} w_i + \delta^2 \underbrace{\sum_{i\in I} w_i \cdot \left(9\alpha^2 \langle A_i, y\rangle^2 - \frac{\beta}{4}\sum_{j=1}^n y_j^2 A_{ij}^2\right)}_{\le 0 \text{ since } y\in U_5^{(t)}} \\ &+ \underbrace{\beta\delta^2 \sum_{i\in I} w_i \cdot \left(36\beta \left\langle A_i^{\circ 2}\circ x, y\right\rangle^2 - \frac{1}{4}\sum_{j=1}^n y_j^2 A_{ij}^2\right)}_{\le 0 \text{ since } y\in U_6^{(t)}} \le \sum_{i\in I} w_i.\end{aligned}$$

This proves the claim. □

2.4.1 The discrepancy guarantee

We can now prove that the algorithm indeed finds a vector satisfying the desired discrepancy bound:

Lemma 20. *For a proper choice of $\alpha := \Theta(\sqrt{\log n})$ and $\beta := \Theta(\log n)$, the algorithm returns a vector $x := x^{(T)}$ with $\langle A_i, x\rangle \le O(\sqrt{\log n})$ for each row $i \in [m]$.*

Proof. First consider a light index $i \in I_{\text{light}}$. The potential function never increases, hence

$$e^{\alpha\langle A_i, x\rangle} \le \Phi^{(T)} \le \Phi^{(0)} \le m \cdot e^{\beta\cdot C} \le n^2 \cdot e^{\beta\cdot C}$$

Taking logarithms and dividing by α gives

$$\langle A_i, x\rangle \le \frac{2\log(n)}{\alpha} + \frac{C\beta}{\alpha} \le (C^2+2)\cdot\sqrt{\log(n)}.$$

Here the last inequality follows for choices of if $\alpha := \sqrt{\log(n)}$ and $\beta := C\log(n) = C\alpha^2$.

Now consider a heavy index i. Let t be the last iteration when $\sum_{j\in A^{(t)}} A_{ij}^2 > C$. Until this point one has $\left\langle A_i, x^{(t)}\right\rangle = 0$. Since $|A_{ij}| \ge 1/(C^2\sqrt{\log n})$ for every non-zero entry, one has

$$|\{j \in A^{(t+1)} : A_{ij} \ne 0\}| \le C^5 \log(n).$$

Hence, regardless how those elements are colored, one has

$$\left\langle A_i, x^{(T)}\right\rangle = \left\langle A_i, x^{(T)} - x^{(t)}\right\rangle \le 2 \sum_{j\in A^{(t+1)}} |A_{ij}| \le O(\sqrt{\log n}).$$

□

2.4.2 Quadratic error in subspaces

It remains to prove that the subspaces $U_5^{(t)}$ and $U_6^{(t)}$ used in the algorithm exist with high enough dimensions. We will prove two lemmas that we keep general:

Lemma 21. *Let $A, B \in \mathbb{R}^{m\times n}$ be any matrices with $|B_{ij}| \le |A_{ij}|$ for all $(i,j) \in [m]\times[n]$ and let $w_1, \dots, w_m \ge 0$ be any weights. Then for any $k \in \mathbb{N}$, one can compute a subspace U of dimension at least $\dim(U) \ge (1-\frac{1}{k})n$ in time $O(n^2(m+n))$ so that*

$$\sum_{i=1}^m w_i \cdot (B_i B_i^T \bullet yy^T) \le k \sum_{i=1}^m w_i \cdot (\mathrm{diag}(A_i^{\circ 2}) \bullet yy^T) \qquad \forall y \in U.$$

Proof. Consider the matrix $L := \sum_{i=1}^m w_i B_i B_i^T$ and $R := k \cdot \sum_{i=1}^m w_i \cdot \mathrm{diag}(A_i^{\circ 2})$. Then the goal is to find a subspace U so that

$$(L \bullet yy^T) \le (R \bullet yy^T)$$

for all $y \in U$.

First, if we replace $A_i' := \sqrt{w_i} A_i$ and $B_i' := \sqrt{w_i} B_i$, then the assumption $|B_{ij}| \le |A_{ij}|$ is preserved and the claim is not changed. Hence we may assume that $w_i = 1$ for all $i \in [m]$.

If $A^j = 0$, then also $B^j = 0$ and

$$(L \bullet e_j e_j^T) = 0 = (R \bullet e_j e_j^T),$$

which means that e_j can be added to the subspace. So let us assume that $A^j \neq 0$ for all j.

Next, if we scale a columns A^j and B^j by some scalar s and we scale y_j by $\frac{1}{s}$, then the claim remains invariant. Hence we assume that $\|A^j\|_2 = 1$ for all $j \in [n]$. Then

$$\mathrm{Tr}[L] = \sum_{i=1}^m \|B_i\|_2^2 = \sum_{j=1}^n \|B^j\|_2^2 \overset{|B_{ij}|\le|A_{ij}|}{\le} \sum_{j=1}^n \underbrace{\|A^j\|_2^2}_{=1} = n$$

On the other hand,

$$R = k \sum_{i=1}^m \mathrm{diag}(A_i^{\circ 2}) = k \cdot \mathrm{diag}\Big(\Big(\underbrace{\sum_{i=1}^m A_{ij}^2}_{=1}\Big)_{j\in[n]}\Big) = k \cdot I$$

Then L must have less than $\frac{n}{k}$ eigenvalues of value more than k. Then we can define U as the span of the eigenvectors of L that have eigenvalue at most k.

Finally, we observe that computing the matrices L, R takes time $O(mn^2)$ and the eigen-decomposition can be done in time $O(n^3)$. □

The existence of the subspace $U_5^{(t)}$ follows from choosing $B := A$ with $k := 16$ and $\frac{\beta}{16\alpha^2} \geq 16$.

The second lemma that we need is the following:

Lemma 22. *Let $A \in [-\gamma, \gamma]^{m \times n}$ and $x \in [-1, 1]^n$. Then for any $k \in \mathbb{N}$ one can compute a subspace $U \subseteq \mathbb{R}^n$ with $\dim(U) \geq n \cdot (1 - \frac{1}{k})$ in time $O(n^2(m+n))$ so that*

$$\sum_{i=1}^{m} w_i \cdot \left\langle (A_i^{\circ 2} \circ x)^{\otimes 2}, yy^T \right\rangle \leq k \cdot \gamma^2 \cdot \sum_{i=1}^{m} w_i \cdot \left\langle \mathrm{diag}(A_i^{\circ 2}), yy^T \right\rangle \qquad \forall y \in U.$$

Proof. We define a matrix $B \in \mathbb{R}^{m \times n}$ by letting $B_i := \frac{A_i^{\circ 2} \circ x}{\gamma}$. Then $|B_{ij}| \leq |A_{ij}|$ and applying Lemma 21 gives the claim. □

Then applying Lemma 22 with $\gamma := \frac{1}{C\sqrt{\beta}}$ and $k = 16$ guarantees the subspace $U_6^{(t)}$. For the running time analysis of Theorem 4, one can set $\delta := \Theta(\frac{1}{\sqrt{\log n}})$ and the algorithm only takes $O(n \log(n))$ iterations, each taking time $O(n^2(m+n))$.

Chapter 3

NUMBER BALANCING IS AS HARD AS MINKOWSKI'S THEOREM

3.1 Introduction

One of *six basic NP-complete problems* of Garey and Johnson [17] is the *partition problem* that for a list of numbers $a_1, \ldots, a_n$ asks whether there is a partition of the indices so that the sums of the numbers in both partitions coincide. Partition and related problems like knapsack, subset sum and bin packing are some of the fundamental classical problems in theoretical computer science with numerous practical applications; see for example the textbooks [34, 25] and the article of Mertens [38].

In this chapter, we study a variant called the *number balancing problem* (NBP), where the goal is to find two disjoint subsets $I_1, I_2 \subseteq \{1, \ldots, n\}$ so that the difference

$$\left| \sum_{i \in I_1} a_i - \sum_{i \in I_2} a_i \right|$$

is minimized.

Equivalently, given a vector of numbers $a = (a_1, \ldots, a_n) \in [0, 1]^n$, we want to find a vector of *signs* $x \in \{-1, 0, 1\}^n \setminus \{0\}$ so that

$$|\langle a, x \rangle| = \left| \sum_{i=1}^{n} x_i a_i \right|$$

is minimized. Woeginger and Yu [51] studied this problem under the name "equal-subset-sum" and showed that it is NP-hard to decide whether there are two disjoint subsets that sum up to the exact same value. This version has also been extensively studied in combinatorics [32, 12, 27].

On the positive side, it is not hard to prove that there is always a solution with exponentially small error. Suppose that $a_1, \ldots, a_n \in [0,1]$. Consider the list of 2^n many numbers $\sum_{i=1}^n a_i x_i$ for all $x \in \{0,1\}^n$. All these numbers fall into the interval $[0,n]$, hence by the *pigeonhole principle*, we can find two distinct vectors $x, x' \in \{0,1\}^n$ with

$$\left| \sum_{i=1}^n a_i x_i - \sum_{i=1}^n a_i x'_i \right| \leq \frac{n}{2^n - 1}.$$

Then $x - x'$ gives the desired solution.

Note that the bound can be slightly improved to $O(\frac{\sqrt{n}}{2^n})$ by using the fact that due to *concentration of measure* effects, for a constant fraction of vectors $x \in \{0,1\}^n$, the sums $\sum_{i=1}^n a_i x_i$ fall into an interval of length $\sqrt{n}$ (instead of n).

However, since these arguments rely on the pigeonhole principle, they are non-constructive. Restricting the non-constructive argument to polynomially many "pigeons" provides a simple polynomial time algorithm to find at least an $x \in \{-1,0,1\}^n \setminus \{0\}$ with

$$|\langle a, x \rangle| \leq \frac{1}{\text{poly}(n)}$$

for an arbitrarily small polynomial.

Interestingly, the only known polynomial time algorithm that gives a better guarantee is Karmarkar and Karp's *differencing algorithm* [24] which provides the bound

$$|\langle a, x \rangle| \leq n^{-c \log(n)}$$

for some constant $c > 0$. Their algorithm uses a recursive scheme; find $\Theta(n)$ pairs of numbers a_i of distance at most $\Theta(\frac{1}{n})$ and create an instance consisting of their differences, then recurse.

This leads to the natural question: *Given $a_1, \ldots, a_n \in [0,1]$, what upper bound on $|\sum_{i=1}^n a_i x_i|$ can be guaranteed if $x \in \{-1,0,1\}^n \setminus \{0\}$ is to be chosen in polynomial time?*

While answering this question directly seems out of reach, we note that NBP falls into the class PPP [44], where good solutions are known to exist due to the pigeonhole principle. It is reasonable therefore to study the relationship between this problem and other problems in PPP.

Recall that given linearly independent vectors $b_1, \ldots, b_n \in \mathbb{R}^n$, a *(full rank) lattice* is the set

$$\Lambda := \left\{ \sum_{i=1}^{n} \lambda_i b_i : \lambda_i \in \mathbb{Z} \; \forall i = 1, \ldots, n \right\}.$$

The set $\{b_1, \ldots, b_n\}$ is called a *basis* for Λ and we define

$$\det(\Lambda) := |\det(B)|.$$

For any lattice $\Lambda \subseteq \mathbb{R}^n$ with $\det(\Lambda) \geq 1$, *Minkowski's Theorem* tells us that any symmetric convex body $K \subseteq \mathbb{R}^n$ of volume at least 2^n must intersect $\Lambda \setminus \{0\}$, see for example [36]. This theorem is proven by placing translates of $\frac{1}{2}K$ at any lattice point and then inferring an overlap due to the pigeonhole principle. Again, one can consider the algorithmic question: *given a symmetric convex body K with volume at least 2^n, for what factor ρ can one be guaranteed to find an $x \in (\rho K) \cap \mathbb{Z}^n \setminus \{0\}$ in polynomial time?*

We would like to point out that this factor ρ is within a polynomial factor of the approximability of the Shortest Vector Problem (SVP), the problem of finding the shortest[1] nonzero vector in a lattice. One direction follows from the fact K can be sandwiched between two ellipsoids that differ by a factor of $\sqrt{n}$ [23]. In the other direction, a reduction of Lenstra and Schnorr shows that given a polynomial-time oracle to find a vector of length at most $f(n)$ in a lattice with $\det(\Lambda) \leq 1$, there is a polynomial-time algorithm to find a vector within $f(n)^2$ of the shortest vector [29]. This is a nontrivial reduction that uses the assumed oracle both on the original lattice and on its dual. Note that Minkowski's theorem guarantees the existence of a lattice vector of length at most $O(\sqrt{n})$.

The complexity of SVP is of great theoretical and practical interest. As a rarity in theoretical computer science, SVP admits (NP$\cap$coNP)-certificates for a value that is at most a factor $O(\sqrt{n})$ away from the optimum [1], while the best known hardness under reasonable complexity assumptions lies at a subpolynomial bound of $n^{\Theta(1/\log\log n)}$ [21]. The famous LLL-algorithm [26] can find a $2^{n/2}$-approximation in polynomial time (the generalized *block*

[1]SVP can be defined for any norm, but anywhere the norm is not specified we consider the Euclidean norm.

reduction method of Schnorr [46] brings the factor down to $2^{n \log\log(n)/\log(n)})$. On the other hand, a polynomial factor approximation of SVP would be enough to break lattice-based cryptosystems [2, 40].

It is not hard to use an *exact* oracle for Minkowski's Theorem to find a good number balancing solution, since the body

$$K := \left\{ x \in (-2, 2) : \left| \sum_{i=1}^{n} x_i a_i \right| \leq \Theta\left(\frac{n}{2^n}\right) \right\}$$

has volume at least 2^n. However, it is not clear how we could use an *approximate* oracle. For example, it is known that the LLL-algorithm can be used to find a nonzero integer vector $x \in \rho K$ for a factor of $\rho = \text{poly}(n) \cdot 2^{n/2}$. While the error guarantee of

$$\left| \sum_{i=1}^{n} a_i x_i \right| \leq \text{poly}(n) \cdot 2^{-n/2}$$

outperforms the Karmarkar-Karp algorithm, we only know that $\|x\|_\infty < 2\rho$, which means that x will not be a valid solution if $\rho > 1$.[2] This leads us to the next question: *what factor ρ is needed for Minkowski's Theorem to improve over Karmarkar-Karp's bound*?

We have seen that in a certain sense NBP can be reduced to an oracle for Minkowski's Theorem, and in fact we will show that there is also a direct reduction to SVP in the ℓ^∞ norm. This brings us to the question about the reverse: *given an oracle that solves NBP within an exponentially small error, can this give a non-trivial oracle for the Shortest Vector Problem or Minkowski's Theorem?*

In this chapter, we provide some answers to the questions raised above, by relating the complexity of the number balancing problem to Minkowski's Theorem and the Shortest Vector Problem. First we give the precise definitions of the problems we will consider.

Definition 1. Suppose $p \in [1, \infty]$ with $B^p(0, 1)$ the closed unit ball in the ℓ^p norm. For $\delta \geq \frac{\Omega(\sqrt{n})}{2^n}$ and $\rho \geq 1$, we define the following problems.

- δ**-NBP**: Given $a \in [0, 1]^n$, find $x \in \{-1, 0, 1\}^n \setminus \{0\}$ with $|\langle a, x \rangle| \leq \delta$.

[2] If $\rho \leq 2 - \varepsilon$, then one can still obtain an error of $|\sum_{i=1}^{n} a_i x_i| \leq 2^{-\Theta(\varepsilon n)}$, but this breaks down if $\rho \geq 2$.

- **ρ-Minkowski Problem**: Given a lattice Λ and a symmetric convex body[3] $K \subseteq \mathbb{R}^n$ with $\mathrm{vol}_n(K) \geq 2^n \det(\Lambda)$, find a vector $x \in (\rho K) \cap \Lambda \setminus \{0\}$.

- **ρ-PromiseSVP$_p$**: Given a lattice $\Lambda \subset \mathbb{R}^n$ with $\mathrm{vol}(B^p(0,1)) \geq 2^n \det(\Lambda)$, find a vector $x \in \Lambda \setminus \{0\}$ with $0 < \|x\|_p \leq \rho$.

As already discussed, δ-NBP and the ρ-Minkowski Problem will always have a solution by nonconstructive arguments. Moreover, we notice that ρ-PromiseSVP$_p$ is just the ρ-Minkowski Problem on an ℓ^p ball, and so it is also guaranteed to have a solution. Notice also that ρ-PromiseSVP$_p$ would also follow immediately from a ρ-approximation to SVP in the ℓ^p norm, since a short enough vector is guaranteed to exist.

We would like to stress that polynomial-time algorithms for any of the three problems are not known to be inconsistent with P $\neq$ NP. The hardness results of SVP do not apply to ρ-PromiseSVP$_p$ since we do not require it to give a shortest vector of the lattice.

We provide the following reduction:

Theorem 23. *Suppose there is a polynomial-time algorithm for the ρ-Minkowski problem for polytopes K with $O(n)$ facets. Then there is a polynomial-time algorithm for δ-NBP where $\delta := 2^{-n^{\Theta(1/\rho)}}$.*

In fact, to obtain an algorithm for δ-NBP, it suffices to have an approximate Minkowski oracle for the linear transformation of a cube, which is equivalent to an oracle for ρ-PromiseSVP$_\infty$.

Theorem 24. *Suppose that there is a polynomial-time algorithm for ρ-PromiseSVP$_\infty$. Then there is a polynomial-time algorithm for δ-NBP, where $\delta := 2^{-n^{\Theta(1/\rho)}}$.*

In particular, an oracle for $\rho \leq c' \log(n)/\log\log(n)$ would imply an improvement over Karmarkar-Karp's algorithm, where $c' > 0$ is a small enough constant.

---

[3]We assume K is given to us by a separation oracle.

Finally, we can also prove that an oracle with exponentially small error for number balancing would provide an approximation for Minkowski's problem:

Theorem 25. *Suppose that there is a polynomial-time algorithm for δ-NBP with $\delta \leq 2^{\sqrt{n}}/2^n$. Then there is a polynomial-time algorithm for the ρ-Minkowski problem for $\rho = O(n^5)$. Here it suffices to have a separation oracle for the convex body $K \subseteq \mathbb{R}^n$.*

3.2 Reducing Number Balancing to Minkowski's Theorem

In this section we will show how to solve NBP with an oracle for Minkowski's problem. The idea is to consider a hypercube intersected with the constraint $|\langle a, x\rangle| \leq \delta$, and to show that this set has large enough volume. If we have an exact Minkowski oracle, this gives us $x \in \{-1, 0, 1\}^n \setminus \{0\}$ as desired. Here we state a more general version which uses only a ρ-approximate Minkowski oracle, and then show how we can use this more general version to solve NBP with a weaker bound.

Theorem 26. *Suppose we have a polynomial-time algorithm for the ρ-Minkowski problem, and let $k > 0$ be any positive integer. Then, for any $a \in [0,1]^n$, there is a polynomial-time algorithm to find an $x \in \mathbb{Z}^n \setminus \{0\}$ with $\|x\|_\infty \leq k$ and so that $|\langle a, x\rangle| \leq n\left(\frac{\rho}{k+1}\right)^{n-1}$.*

Proof. For ease of notation, let

$$\delta := n\left(\frac{\rho}{k+1}\right)^{n-1}.$$

Now consider the body

$$K := \left\{x \in \left(-\frac{k+1}{\rho}, \frac{k+1}{\rho}\right)^n : |\langle a, x\rangle| \leq \delta\right\}.$$

Obviously this is a symmetric convex body. For $\alpha \in [-\delta, \delta]$, consider the $(n-1)$-dimensional slice

$$K(\alpha) := \left\{x \in \left(-\frac{k+1}{\rho}, \frac{k+1}{\rho}\right)^n : \langle a, x\rangle = \alpha\right\}$$

of it. Let us consider the $(n-1)$-dimensional volume $\mathrm{vol}_{n-1}(K(\alpha))$ of that slice. Then we can write the volume of K as

$$\mathrm{vol}_n(K) = \int_{-\delta}^{\delta} \mathrm{vol}_{n-1}(K(\alpha))\, d\alpha.$$

Moreover

$$\left(\frac{2(k+1)}{\rho}\right)^n = \mathrm{vol}_n\left(-\frac{k+1}{\rho}, \frac{k+1}{\rho}\right)^n = \int_{-\frac{n(k+1)}{\rho}}^{\frac{n(k+1)}{\rho}} \mathrm{vol}_{n-1}(K(\alpha))\, d\alpha.$$

since the slices are empty if $|\alpha| > \frac{n(k+1)}{\rho}$. By symmetry $\mathrm{vol}_{n-1}(K(\alpha)) = \mathrm{vol}_{n-1}(K(-\alpha))$. Moreover, by convexity of K, for $\alpha \geq 0$, the quantity $\mathrm{vol}_{n-1}(K(\alpha))$ is monotonically non-increasing. Thus

$$\begin{aligned}\mathrm{vol}_n(K) &= \int_{-\delta}^{\delta} \mathrm{vol}_{n-1}(K(\alpha))\, d\alpha \geq \frac{\delta}{\left(\frac{n(k+1)}{\rho}\right)} \cdot \int_{-\frac{n(k+1)}{\rho}}^{\frac{n(k+1)}{\rho}} \mathrm{vol}_{n-1}(K(\alpha))\, d\alpha \\ &= \frac{\delta}{\left(\frac{n(k+1)}{\rho}\right)} \cdot \left(\frac{2(k+1)}{\rho}\right)^n.\end{aligned}$$

Now, using the fact that $\delta = n\left(\frac{\rho}{(k+1)}\right)^{n-1}$, we get $\mathrm{vol}_n(K) \geq 2^n$. Hence, using our ρ-approximate Minkowski oracle, we can find a vector

$$x \in (\rho K \cap \mathbb{Z}^n) \setminus \{0\} = ((-(k+1), (k+1))^n \cap \mathbb{Z}^n) \setminus \{0\}.$$

In particular, this gives $x \in \mathbb{Z}^n$ with $|\langle a, x\rangle| \leq n\left(\frac{\rho}{k+1}\right)^{n-1}$ and $\|x\|_\infty \leq k$. □

The bound in Theorem 26 could be strengthened by a $\sqrt{n}$ factor by using concentration of measure arguments. We omit this here.

Suppose now, for instance, that $\rho = (2 - \epsilon)$ for some $\epsilon \in (0, 1]$. Then we can pick $k = 1$ and get $|\langle a, x\rangle| \leq 2^{-\Theta(\epsilon n)}$ with $\|x\|_\infty \leq 1$ and $x \in \mathbb{Z}^n \setminus \{0\}$. However, this line of arguments breaks down for $\rho \geq 2$ as these would in general not produce feasible solutions for number balancing.

Instead of using an oracle for the ρ-Minkowski Problem one can directly use an oracle for ρ-PromiseSVP$_\infty$. We need the following theorem:

Theorem 27. *Suppose that there is a polynomial-time algorithm for ρ-PromiseSVP$_\infty$. Then for any $a \in [0, 1]^n$ there is a polynomial-time algorithm to find $x \in \mathbb{Z}^n \setminus \{0\}$ with $\|x\|_\infty \leq k$ and so that $|\langle a, x\rangle| \leq 2nk\rho(\frac{\rho}{k})^n$.*

Proof. When $\rho(\frac{\rho}{k})^n \geq \frac{1}{2}$, we have $2nk\rho(\frac{\rho}{k})^n \geq 1$, in which case $x = (1, 0, \cdots, 0)$ trivially does the job. We can hence assume that $\rho(\frac{\rho}{k})^n < \frac{1}{2}$.

Let I_n be the $n \times n$ identity matrix and consider the lattice Λ generated by the $(n+1) \times (n+1)$-dimensional matrix

$$B := \begin{pmatrix} \frac{\rho}{k} I_n & 0 \\ \frac{1}{2nk}(\frac{k}{\rho})^n a^T & (\frac{k}{\rho})^n \end{pmatrix}.$$

Note that $\det(B) = 1$. Let $x \in \Lambda$ be the vector returned by the algorithm. Then $y := B^{-1}x \in \mathbb{Z}^{n+1}$. Since $\|x\|_\infty \leq \rho$ and $x = By$, we have $|y_i| \leq k$ for $i = 1, \ldots, n$. Moreover,

$$\begin{aligned} \rho \geq |x_{n+1}| &= \left| y_{n+1} \left(\frac{k}{\rho}\right)^n + \sum_{i=1}^{n} \frac{1}{2nk} \left(\frac{k}{\rho}\right)^n y_i a_i \right| \\ &\geq |y_{n+1}| \left(\frac{k}{\rho}\right)^n - \left| \sum_{i=1}^{n} \frac{1}{2nk} \left(\frac{k}{\rho}\right)^n y_i a_i \right| \\ &\overset{|y_i| \leq k}{\geq} |y_{n+1}| \left(\frac{k}{\rho}\right)^n - \frac{1}{2} \left(\frac{k}{\rho}\right)^n, \end{aligned}$$

which implies

$$|y_{n+1}| \leq \frac{1}{2} + \rho(\frac{\rho}{k})^n.$$

Since $\rho(\frac{\rho}{k})^n < \frac{1}{2}$ and $y_{n+1} \in \mathbb{Z}$, we must have $y_{n+1} = 0$.

Therefore,

$$\left| \sum_{i=1}^{n} \frac{1}{2nk} \left(\frac{k}{\rho}\right)^n y_i a_i \right| = |x_{n+1}| \leq \rho,$$

and hence

$$\left| \sum_{i=1}^{n} y_i a_i \right| \leq 2nk\rho \left(\frac{\rho}{k}\right)^n,$$

so the vector $(y_1, \ldots, y_n)$ does the job.

□

However, we still face a problem in the case $\rho \geq 2$. It turns out that we can design a *recursive self-reduction* to allow us to use larger ρ. The main technical argument is to transform an algorithm that finds $x \in \mathbb{Z}^n \setminus \{0\}$ with $\|x\|_\infty \leq k$ for $k \geq 2$ into an algorithm that finds vectors $x \in \mathbb{Z}^n \setminus \{0\}$ with $\|x\|_\infty \leq \frac{k}{2}$, with a bounded decay in the error $|\langle a, x \rangle|$. Applying this recursively gives the following lemma.

Lemma 28. *Suppose that there is a polynomial-time algorithm that for any $a' \in [0,1]^n$ finds a vector $x' \in \{-k,\dots,k\}^n \setminus \{0\}$ with $|\langle a', x'\rangle| \le 2^{-n}$. If $k \le \frac{\log n}{6 \log\log n}$, then there is also a polynomial-time algorithm that for any $a \in [0,1]^n$ finds a vector $x \in \mathbb{Z}^n$ with $|\langle a, x\rangle| \le 2^{-n^{\frac{1}{3k}}}$ and $x \in \{-1,0,1\}^n \setminus \{0\}$.*

Before we go through the self reduction, we show how Lemma 28 gives Theorems 23 and 24.

Proof of Theorems 23 and 24. If $\rho \ge \frac{\log n}{48 \log\log n}$, then $2^{-n^{\Theta(1/\rho)}} = 2^{-\log^{O(1)} n}$. By choosing a proper constant on the exponent, this can be achieved with the Karmarkar-Karp algorithm. So we only need to work with $\rho < \frac{\log n}{48 \log\log n}$.

Now suppose we have a polynomial-time algorithm for the ρ-Minkowski Problem (resp. ρ-PromiseSVP$_\infty$). If we take $k = 3\rho$, Theorem 26 (resp. Theorem 27) gives a polynomial-time algorithm to find x with $\|x\|_\infty \le k$ and $|\langle a, x\rangle| \le 2^{-n}$.

Moreover, $k = 3\rho \le \frac{\log n}{16 \log\log n}$, and hence the condition of Lemma 28 is satisfied. Then the bound given by Lemma 28 is $2^{-n^{\frac{1}{3k}}} \le 2^{-n^{\Theta(1/\rho)}}$. □

We now prove Lemma 28. The way we do the self-reduction is the following. We partition our set of n numbers into subsets of size $\sqrt{n}$. First, for each subset ℓ, we find a number $b_\ell \neq 0$ for which we can (approximately) express $b_\ell, 2b_\ell, \dots, kb_\ell$ as linear combinations of elements of that subset using only coefficients in $\{-\lfloor \frac{k}{2} \rfloor, \dots \lfloor \frac{k}{2} \rfloor\}$. We then run our assumed algorithm on $b_1, \dots, b_{\sqrt{n}}$ to obtain $y \in \{-k,\dots,k\}^n$ with $\langle b, y\rangle = \sum_{\ell=1}^{\sqrt{n}} y_\ell b_\ell$ being small. Since each of the summands can be expressed more efficiently in terms of our original set of numbers, we obtain a good solution x with coefficients in $\{-\lfloor \frac{k}{2} \rfloor, \dots \lfloor \frac{k}{2} \rfloor\}$.

The following two lemmas go through this argument more precisely. Note that the interesting parameter choice is $r := \lceil k/2 \rceil$, so that the size of the coefficients is halved.

Lemma 29. *Let $r, k \in \mathbb{N}$ be parameters with $0 < r < k$ and let $\delta \ge 0$. Let $\alpha_1, \dots, \alpha_k \in \mathbb{R}$ so that $|\sum_{i=1}^k i \cdot \alpha_i| \le \delta$ and abbreviate $\beta := \alpha_r + \dots + \alpha_k$. Then for any $j \in \{0,\dots,k\}$ one*

can find coefficients $\lambda_{i,j} \in \mathbb{Z}$ *with*

$$|\lambda_{i,j}| \leq \max\{r-1, k-r\}$$

and

$$|j \cdot \beta - \sum_{i=1}^{k} \lambda_{i,j}\alpha_i| \leq \delta.$$

Proof. By symmetry it suffices to consider $j \geq 0$. For $j \in \{0, \ldots, r-1\}$, we can obviously write

$$j \cdot \beta = j \cdot \alpha_r + \ldots + j \cdot \alpha_k.$$

Now consider $j \in \{r, \ldots, k\}$. The trick is to use the fact that

$$j \cdot \beta - \sum_{i=1}^{k} i\alpha_i = \sum_{i=r}^{k} j \cdot \alpha_i - \sum_{i=1}^{k} i\alpha_i = \sum_{i=r}^{k}(j-i)\alpha_i + \sum_{i=1}^{r-1}(-i)\alpha_i.$$

If we inspect the size of the used coefficients, then for $i \in \{1, \ldots, r-1\}$ we have $|-i| \leq r-1$ and for $i \in \{r, \ldots, k\}$ we have $|j-i| \leq k-r$. □

Lemma 30. *Let* $k, r \in \mathbb{N}$ *be parameters with* $0 < r < k$. *Let* $f : \mathbb{N} \to \mathbb{R}$ *be a non-negative function such that* $f(n) \geq 4 \log n$. *Suppose that there is a polynomial-time algorithm that for any* $a' \in [0,1]^n$ *finds a vector* $x' \in \{-k, \ldots, k\}^n \setminus \{0\}$ *with* $|\langle a', x' \rangle| \leq 2^{-f(n)}$. *Then there is also a polynomial-time algorithm that for any* $a \in [0,1]^n$ *finds a vector* $x \in \mathbb{Z}^n \setminus \{0\}$ *with*

$$|\langle a, x \rangle| \leq 2^{-2f(\lfloor\sqrt{n}\rfloor)/3}$$

and

$$\|x\|_\infty \leq \max\{r-1, k-r\}.$$

Proof. Let $a \in [0,1]^n$ be the given vector of numbers. To keep notation simpler, we will assume n is a perfect square. If it is not, we can replace $\sqrt{n}$ by $\lfloor\sqrt{n}\rfloor$.

Split $[n]$ into blocks $I_1, \ldots, I_{\sqrt{n}}$ each of size $|I_\ell| = \sqrt{n}$. For each block I_ℓ we use the oracle to find a vector $x_\ell \in \{-k, \ldots, k\}^n \setminus \{0\}$ with $\mathrm{supp}(x_\ell) \subseteq I_\ell$ so that $|\langle a, x_\ell \rangle| \leq 2^{-f(\sqrt{n})}$.

If for any ℓ one has $\|x_\ell\|_\infty \leq r-1$, then we simply return $x := x_\ell$ and are done. Otherwise, we write the vector as $x_\ell = \sum_{i=1}^{k} i \cdot x_{\ell,i}$ with vectors $x_{\ell,1}, \ldots, x_{\ell,k} \in \{-1, 0, 1\}^n$. Note that these vectors will have disjoint support and $\mathrm{supp}(x_{\ell,1}), \ldots, \mathrm{supp}(x_{\ell,k}) \subseteq I_\ell$. Moreover we know that for every ℓ there is at least one index $i \in \{r, \ldots, k\}$ with $x_{\ell,i} \neq 0$.

Now define a vector $b \in \mathbb{R}^{\sqrt{n}}$ with $b_\ell := \sum_{i=r}^{k} \langle a, x_{\ell,i} \rangle$. Note that if for any ℓ we have $|b_\ell| \leq 2^{-f(\sqrt{n})}$, then we can set $x = \sum_{i=r}^{k} x_{\ell,i}$ and we are done. Therefore we may assume that $|b_\ell| > 2^{-f(\sqrt{n})}$ for all ℓ.

Also note that since the $x_{\ell,i}$ have disjoint support, we have $\|b\|_\infty \leq \sqrt{n}$. We run the oracle again to find a vector $y \in \{-k, \ldots, k\}^{\sqrt{n}} \setminus \{0\}$ so that $| \langle b, y \rangle | \leq \sqrt{n} \cdot 2^{-f(\sqrt{n})}$. For each block $\ell \in [\sqrt{n}]$ we can use Lemma 29 to find integer coefficients $\lambda_{\ell,i}$ with $|\lambda_{\ell,i}| \leq \max\{r-1, k-r\}$ so that

$$\left| y_\ell \cdot b_\ell - \sum_{i=1}^{k} \lambda_{\ell,i} \cdot \langle a, x_{\ell,i} \rangle \right| \leq 2^{-f(\sqrt{n})}.$$

We define

$$x := \sum_{\ell=1}^{\sqrt{n}} \sum_{i=1}^{k} \lambda_{\ell,i} x_{\ell,i}.$$

Then $\|x\|_\infty \leq \max\{r-1, k-r\}$ since the $x_{\ell,i}$'s have disjoint support and $\|x_{\ell,i}\|_\infty \leq 1$ for all ℓ, i. Moreover, since there is some $y_\ell \neq 0$ and $|b_\ell| > 2^{-f(\sqrt{n})}$, we have $x \neq 0$.

Finally we inspect that

$$\begin{aligned} | \langle a, x \rangle | &\leq | \langle y, b \rangle | + \sum_{\ell=1}^{\sqrt{n}} \left| y_\ell b_\ell - \sum_{i=1}^{k} \lambda_{\ell,i} \langle a, x_{\ell,i} \rangle \right| \\ &\leq 2\sqrt{n} \cdot 2^{-f(\sqrt{n})} \leq 2^{-2f(\sqrt{n})/3}. \end{aligned}$$

The last line comes from the fact that when $f(n) \geq 4 \log n$, we have

$$2\sqrt{n} \leq 2^{\frac{2}{3} \log n} = 2^{\frac{4}{3} \log \sqrt{n}} \leq 2^{\frac{1}{3} f(\sqrt{n})}.$$

□

Now we can apply Lemma 30 recursively to prove Lemma 28.

Proof of Lemma 28. Suppose $k \leq \frac{\log n}{6 \log \log n}$ and set $r = \lceil k/2 \rceil$. Consider the function

$$f_t(n) = 2^{-t} n^{2^{-t}}$$

for $t = 0, \ldots, \lceil \log k \rceil$, and notice that

$$2^{-t} \geq \frac{1}{2k} \geq 2 \frac{\log \log n}{\log n}.$$

Therefore we have

$$f_t(n) = 2^{-t} n^{2^{-t}} \geq \frac{1}{2k} n^{1/2k} \geq \left(2 \frac{\log \log n}{\log n} \right) \cdot \log^2 n \geq 4 \log n.$$

Finally, notice that

$$\frac{2}{3} f_t(\lfloor \sqrt{n} \rfloor) \geq \frac{1}{2} f_t(\sqrt{n}) = f_{t+1}(n).$$

We are now able to apply Lemma 30. In particular, suppose we have an oracle to find x with $\|x\|_\infty \leq 2^{-t} k$ and $|\langle a, x \rangle| \leq 2^{-f_t(n)}$. Then Lemma 30 gives us an oracle to find x with

$$\|x\|_\infty \leq 2^{-(t+1)} k$$

and

$$|\langle a, x \rangle| \leq 2^{-f_{t+1}(n)}.$$

(Here we ignore the dependence of t on n - notice that t is nondecreasing in n, so replacing $t(n)$ by $t(\sqrt{n})$ only increases $f_t(n)$.)

Running this $\lceil \log k \rceil \leq \log 2k$ times gives a bound of

$$2^{-f_{\log 2k}(n)} = 2^{-n^{1/2k}/2k} \leq 2^{-n^{1/3k}}.$$

Here we use the fact that when $k \leq \frac{1}{6} \frac{\log n}{\log \log n}$, we have $\frac{n^{\frac{1}{2k}}}{2k} \geq n^{\frac{1}{3k}}$. □

3.3 Reducing Minkowski's Theorem to Number Balancing

In this section we show that for small enough δ, an oracle for δ-NBP can be used to design an algorithm for ρ-Minkowski's Problem, where ρ is polynomial in n.

The first helpful insight is that any symmetric convex body can be approximated within a factor of $\sqrt{n}$ using an *ellipsoid* [30, 20]. Recall that an ellipsoid is a set of the form

$$\mathcal{E} = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} \frac{1}{\lambda_i^2} \cdot \langle x, a_i \rangle^2 \leq 1 \right\} \tag{3.1}$$

with an *orthonormal basis* $a_1, \ldots, a_n \in \mathbb{R}^n$ defining the *axes* and positive coefficients $\lambda_1, \ldots, \lambda_n > 0$ that describe the *lengths of the axes*[4]. Overall, our reduction will operate in two steps:

(i) By combining *John's Theorem* with *lattice basis reduction*, we can show that it suffices to find integer points in an ellipsoid that is *well-rounded*, meaning that the lengths of the axes are bounded.

(ii) We show that a number balancing oracle allows a self-reduction to a generalized form where inner products with n vectors have to be minimized and additionally the solution space is $\mathbb{Z}^n$ instead of $\{-1, 0, 1\}^n$.

We begin by proving (ii) and postpone (i) until the end of this section.

3.3.1 A self-reduction to a generalized form of number balancing

Recall that for $\delta > 0$, we defined δ-NBP as follows: given $a \in [0, 1]$, find a vector $x \in \{-1, 0, 1\}^n \setminus \{0\}$ with $|\langle a, x \rangle| \leq \delta$. Notice that we may allow for vectors $a \in [-1, 1]^n$ without changing the problem, as one can flip the signs of x as needed to accommodate for the changes in sign of a.

The main technical result of this section is the following reduction:

[4]Strictly speaking, the length of axis i is $2\lambda_i$, but we will continue calling λ_i the "axis length".

Theorem 31. *Suppose there is a polynomial-time algorithm for δ-NBP with $\delta = \frac{g(n)}{2^n}$ and $g(n) \leq 2^{n/2}$. Then there is a polynomial-time algorithm that on input $a_1, \ldots, a_n \in [-1,1]^n$ and $0 < \lambda_1 \leq \ldots \leq \lambda_n \leq 2^n$ with $\prod_{i=1}^n \lambda_i \geq 1$, finds a vector $x \in \mathbb{Z}^n \setminus \{0\}$ with*

$$|\langle x, a_i \rangle| \leq O(n^4) \cdot \lambda_i \cdot g(4n^2)^{1/n} \quad \forall i = 1, \ldots, n.$$

In particular if $g(n) \leq 2^{\sqrt{n}}$, then the right hand side in Theorem 31 simplifies to just $O(n^4) \cdot \lambda_i$. We will show this by introducing two extensions of the number balancing oracle. The first extension gives a weaker bound in terms of the error parameter, but allows for multiple vectors in $[-1,1]^n$. In the second extension, we extend the range of coefficients from $\{-1,0,1\}$ to $\{-Q, \ldots, Q\}$ which leads to a much stronger error bound.

Lemma 32. *Suppose there is a polynomial-time algorithm for δ-NBP. Then there is a polynomial-time algorithm that given an input $a_1, \ldots, a_k \in [-1,1]^n$ and parameters $\delta_1, \ldots, \delta_k \leq \frac{1}{2}$ with $\prod_{i=1}^k \delta_i \geq \delta$ finds a vector $x \in \{-1,0,1\}^n \setminus \{0\}$ with $|\langle a_i, x \rangle| \leq 2n^2 \delta_i$ for all $i = 1, \ldots, k$.*

Proof. The idea is that we will discretize all of the vectors and then run our oracle on their sum. The vector that we obtain will then have small inner product with all of the a_i. To define the discretization $\tilde{a}_i$, round elements of a_i down to the nearest multiple of $2n\delta_i$, and then multiply by $\prod_{j<i} \delta_j$. Defining $\tilde{a}_i$ this way, notice that for all i we have $|\langle \tilde{a}_i, x \rangle| \leq n \prod_{j<i} \delta_j$ for any x satisfying $\|x\|_\infty \leq 1$.

Now let

$$c = \tilde{a}_1 + \ldots + \tilde{a}_k.$$

By our oracle, we can find $x \in \{-1,0,1\}^n \setminus \{0\}$ with $|\langle c, x \rangle| \leq \delta$. Recall that

$$\delta \leq \prod_{i=1}^k \delta_i \leq n \prod_{j \leq k} \delta_j.$$

We have

$$\begin{aligned} |\langle \tilde{a}_1, x \rangle| &\leq |\langle \tilde{a}_2, x \rangle| + \ldots + |\langle \tilde{a}_k, x \rangle| + |\langle c, x \rangle| \leq n\delta_1 + n \prod_{j \leq 2} \delta_j + \ldots + n \prod_{j \leq k} \delta_j \\ &\leq n\delta_1 \cdot \left(1 + \frac{1}{2} + \ldots + \frac{1}{2^k}\right) < 2n\delta_1. \end{aligned}$$

Therefore $|\langle \tilde{a}_1, x\rangle| = 0$. Similarly, for $1 < i \leq k$, if $|\langle \tilde{a}_1, x\rangle|, \ldots, |\langle \tilde{a}_{i-1}, x\rangle| = 0$, then we have

$$|\langle \tilde{a}_i, x\rangle| \leq |\langle \tilde{a}_{i+1}, x\rangle| + \ldots + |\langle \tilde{a}_k, x\rangle| + |\langle c, x\rangle| < 2n \prod_{j \leq i} \delta_j,$$

and hence $|\langle \tilde{a}_i, x\rangle| = 0$ for all i.

Notice that by definition of $\tilde{a}_i$ we have

$$\left\| \left(\prod_{j<i} \delta_j \right) a_i - \tilde{a}_i \right\|_\infty \leq 2n \prod_{j \leq i} \delta_j.$$

Therefore

$$\left| \langle \prod_{j<i} \delta_j a_i, x\rangle \right| \leq 2n^2 \prod_{j \leq i} \delta_j,$$

and so we can conclude that

$$|\langle a_i, x\rangle| \leq 2n^2 \delta_i.$$

□

Now we come to a second reduction that takes the oracle constructed in Lemma 32 as a starting point:

Lemma 33. *Assume there exists a polynomial-time algorithm for δ-NBP where $\delta = f(n)$. Let $a_1, \ldots, a_k \in [-1, 1]^n$ be given with parameters $\delta_1, \ldots, \delta_k \leq \frac{1}{2}$ and a number Q that is a power of 2 and satisfies $\prod_{i=1}^k \delta_i \geq f(n \log Q)$. Then in polynomial time we can find a vector $x \in \{-Q, \ldots, Q\}^n \setminus \{0\}$ with $|\langle a_i, x\rangle| \leq \delta_i Q \cdot 2(n \log Q)^2$ for all $i = 1, \ldots, k$.*

Proof. For each i, we define $b_i \in [-1, 1]^{n \log Q}$ by

$$b_i(j, \ell) = a_i(j) 2^{-\ell}$$

for $j = 1, \ldots, n$ and $\ell = 1, \ldots, \log Q$.

Since

$$\prod_{i=1}^k \delta_i \geq f(n \log Q),$$

we can apply Lemma 32 to find $y \in \{-1,0,1\}^{n \log Q} \setminus \{0\}$ with

$$|\langle b_i, y\rangle| \leq \delta_i \cdot 2(n \log Q)^2.$$

Now define $x \in \{-Q, \ldots, Q\}^n \setminus \{0\}$ by

$$x_j := Q \sum_{\ell=1}^{\log Q} 2^{-\ell} y_{j\ell}.$$

Then for $i = 1, \ldots, k$ we have

$$\delta_i \cdot (2n \log Q)^2 \geq |\langle b_i, y\rangle| = \Big| \sum_{j=1}^{n} \underbrace{\sum_{\ell=1}^{\log Q} y_{j\ell} 2^{-\ell}}_{=x_j/Q} a_i(j) \Big| = \frac{1}{Q} \cdot |\langle a_i, x\rangle|$$

and rearranging gives the claim. □

Finally we come to the proof of Theorem 31.

Proof of Theorem 31. Suppose that the oracle has parameter $f(n) = \rho(n)/2^n$. Suppose that $a_1, \ldots, a_n \in [-1,1]^n$ and $\lambda_1, \ldots, \lambda_n > 0$ with $\prod_{i=1}^n \lambda_i \geq 1$. We choose $Q := 2^{4n}$, which is a power of 2.

Define $\delta_i = \lambda_i \cdot f(n \log Q)^{1/n}$. Note that

$$\delta_i \leq 2^{3n/2} \cdot f(4n^2)^{1/n} \leq \frac{1}{2}$$

since $f(n) \leq 2^{-n/2}$.

Then $\prod_{i=1}^n \delta_i \geq f(n \log Q)$, and so by Lemma 33 we can find $y \in \{-Q, \ldots, Q\}^n \setminus \{0\}$ with

$$\begin{aligned} |\langle a_i, y\rangle| \leq Q\delta_i \cdot 2(n \log Q)^2 &\leq Q\lambda_i \cdot f(n \log Q)^{1/n} \cdot 2(n \log Q)^2 \\ &= \lambda_i \cdot \rho(4n^2)^{1/n} \cdot 2 \cdot (4n^2)^2. \end{aligned}$$

□

3.3.2 A reduction to well-rounded ellipsoids

Using John's Theorem [23], the convex body K in Theorem 25 can be approximated by an ellipsoid $\mathcal{E}$ as defined in Eq. (3.1). The natural approach will then be to apply Theorem 31 to the axes of the ellipsoid. However, it will be crucial that the lengths of the axes of the ellipsoid are bounded by $2^{O(n)}$. We will now argue how to make an arbitrary ellipsoid well rounded.

Let us denote

$$\lambda_{\max}(\mathcal{E}) := \max\{\lambda_i : i = 1, \ldots, n\}$$

as the maximum length of an axis. Recall that a matrix $U \in \mathbb{R}^{n\times n}$ is *unimodular* if $U \in \mathbb{Z}^{n\times n}$ and $|\det(U)| = 1$. In particular, the linear map $T : \mathbb{R}^n \to \mathbb{R}^n$ with $T(x) = Ux$ is a bijection on the integer lattice, meaning that $T(\mathbb{Z}^n) = \mathbb{Z}^n$.

It turns out that one can use the *lattice basis reduction* method by Lenstra, Lenstra and Lovász [26] to find a unimodular linear transformation that "regularizes" any given ellipsoid. Note that it suffices to work with the regularized ellipsoid $T(\mathcal{E})$ since $\mathrm{vol}_n(T(\mathcal{E})) = \mathrm{vol}_n(\mathcal{E})$ and if we find a point $x \in (\rho T(\mathcal{E})) \cap \mathbb{Z}^n$, then by linearity $T^{-1}(x) \in \rho\mathcal{E}$ and $T^{-1}(x) \in \mathbb{Z}^n$.

Given $b_1, \ldots, b_n \in \mathbb{R}^n$ we define the *Gram-Schmidt orthogonalization* iteratively as

$$\hat{b}_j = b_j - \sum_{i<j} \mu_{ij}\hat{b}_i,$$

where

$$\mu_{ij} = \frac{\langle b_j, \hat{b}_i\rangle}{\|\hat{b}_i\|_2^2}.$$

Notice that we can then write

$$b_j = \hat{b}_j + \sum_{i<j} \mu_{ij}\hat{b}_i.$$

In particular, suppose B is the matrix with columns $b_1, \ldots, b_n$ and $\hat{B}$ is the matrix with columns $\hat{b}_1, \ldots, \hat{b}_n$. Then $B = \hat{B}V$ for an upper triangular matrix V with ones along the diagonal and $V_{ij} = \mu_{ij}$ for $i < j$.

Definition 2. Let $B \in \mathbb{R}^{n\times n}$ be a lattice basis and let μ_{ij} be the coefficients from Gram-Schmidt orthogonalization. The basis is called *LLL reduced* if

- (Coefficient-reduced): $|\mu_{ij}| \leq \frac{1}{2}$ for all $1 \leq i < j \leq n$.
- (Lovász condition): $\|\hat{b}_i\|_2^2 \leq 2\|\hat{b}_{i+1}\|_2^2$ for $i = 1, ..., n-1$.

LLL reduction has been widely used in diverse fields such as integer programming and cryptography [42]. One property of the LLL reduced basis is that the eigenvalues of the corresponding matrix B are bounded away from 0:

Lemma 34. *Let B denote the matrix with columns $b_1, \ldots, b_n$. If $b_1, \ldots, b_n$ is an LLL-reduced basis with $\|b_i\|_2 \geq 1$ for all i, then $\|Bx\|_2 \geq 2^{-3n/2} \cdot \|x\|_2$ for all $x \in \mathbb{R}^n$.*

Proof. Let $\hat{B}$ denote the Gram-Schmidt orthogonalization of B, with columns $\hat{b}_1, \ldots, \hat{b}_n$.

Now, for any k, we can use the properties of LLL reduction to gain the following bound (proposition (1.7) in [26]).

$$1 \leq \|b_k\|_2^2 = \|\hat{b}_k\|_2 + \sum_{i<k} \mu_{ik}^2 \|\hat{b}_i\|_2^2 \leq \left(1 + \frac{1}{4}\sum_{i<k} 2^{k-i}\right) \cdot \|\hat{b}_k\|_2^2 \leq 2^k \cdot \|\hat{b}_k\|_2^2.$$

In particular, $\|\hat{b}_k\|^2 \geq 2^{-n}$ for all $k = 1, \ldots, n$. Now let V be the matrix so that $B = \hat{B}V$, and let $x \in \mathbb{R}^n$ with $\|x\|_2 = 1$. Let k denote the largest index with $|x_k| \geq 2^{-k}$. Then

$$|(Vx)_k| = \left|x_k + \sum_{j>k} \mu_{kj} x_j\right| \geq |x_k| - \frac{1}{2}\sum_{j>k} |x_j| \geq 2^{-k} - \frac{1}{2}\sum_{j>k} 2^{-j} \geq 2^{-n}.$$

Now, by the orthogonality of $\hat{b}_1, \ldots, \hat{b}_n$, we have

$$\|Bx\|_2^2 = \|\hat{B}Vx\|_2^2 = \sum_{i=1}^{n} (Vx)_i^2 \|\hat{b}_i\|_2^2 \geq |(Vx)_k|^2 \cdot 2^{-n} \geq 2^{-3n}.$$

Taking square roots gives the claim. □

Lemma 35. *Let $\mathcal{E} = \{x \in \mathbb{R}^n : \|Ax\|_2^2 \leq 1\}$ be an ellipsoid. Then in polynomial time, we can find*

(1) either a vector $x \in \mathcal{E} \cap \mathbb{Z}^n$

(2) or a linear transformation T *so that* $T(x) = Ux$ *for a unimodular matrix* U *and* $\lambda_{max}(T(\mathcal{E})) \leq 2^{3n/2}$.

Proof. Use the algorithm of [26] to find a unimodular matrix U such that $B = AU$ is LLL reduced. Let $b_1, \ldots, b_n$ denote the columns of B. Notice that if $\|b_i\|_2 \leq 1$, then $A^{-1}b_i \in \mathcal{E} \cap \mathbb{Z}^n$, and so we are done. So assume now that $\|b_i\|_2 \geq 1$ for all i.

Define $T(x) = U^{-1}x$, and notice that

$$T(\mathcal{E}) = \{x \in \mathbb{R}^n : \|Bx\|_2^2 \leq 1\}.$$

We then have

$$\lambda_{\max}(T(\mathcal{E})) = \max_{x \in f(\mathcal{E})} \|x\|_2 = \max_{\|Bx\|_2 \leq 1} \|x\|_2 = \max_{x \neq 0} \frac{\|x\|_2}{\|Bx\|_2} \leq 2^{3n/2},$$

where the last inequality follows from Lemma 34. □

Finally we can prove one of our main results, Theorem 25.

Proof of Theorem 25. Let $K \subseteq \mathbb{R}^n$ be a symmetric convex body with $\mathrm{vol}_n(K) \geq 2^n$. We compute an ellipsoid[5]

$$\mathcal{E} = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} \frac{1}{\lambda_i^2} \langle x, a_i \rangle^2 \leq 1 \right\}$$

so that

$$\frac{1}{5\sqrt{n}}\mathcal{E} \subseteq K \subseteq \frac{1}{5}\sqrt{n}\mathcal{E}.$$

Then

$$2^n \cdot 5^n \cdot n^{-n/2} \leq \mathrm{vol}_n(K) \cdot 5^n \cdot n^{-n/2} \leq \mathrm{vol}_n(\mathcal{E}) = \underbrace{\mathrm{vol}(B(0,1))}_{\leq 5^n n^{-n/2}} \cdot \prod_{i=1}^{n} \lambda_i.$$

and hence $\prod_{i=1}^{n} \lambda_i \geq 1$.

[5]Note that there *exists* an ellipsoid that approximates K within a factor of $\sqrt{n}$ and if K is a polytope with m facets, then this ellipsoid can be found in time polynomial in n and m. However, if one only has a separation oracle for K, then the best factor achievable in polynomial time is n [20].

We apply Lemma 35 to either find an integer point in $\mathcal{E}$ and we are done, or we find a unimodular transformation T so that the ellipsoid $\tilde{\mathcal{E}} := T(\mathcal{E})$ has all axes of length at most $2^{O(n)}$.

Suppose the latter case happens. We write

$$\tilde{\mathcal{E}} = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} \frac{1}{\tilde{\lambda}_i^2} \langle x, \tilde{a}_i \rangle^2 \leq 1 \right\}$$

and observe that still $\prod_{i=1}^{n} \tilde{\lambda}_i \geq 1$ as the volume of the ellipsoid has not changed.

We make use of the δ-approximation for the number balancing problem to apply Theorem 31 to the vectors $\tilde{a}_1, \ldots, \tilde{a}_n$ and parameters $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_n$ and obtain a vector $x \in \mathbb{Z}^n \setminus \{0\}$ with

$$| \langle \tilde{a}_i, x \rangle | \leq \tilde{\lambda}_i \cdot O(n^4).$$

Then

$$\sum_{i=1}^{n} \frac{1}{\tilde{\lambda}_i^2} \langle a_i, x \rangle^2 \leq O(n^9)$$

and hence $x \in O(n^{4.5}) \cdot \tilde{\mathcal{E}}$. Then

$$T^{-1}(x) \in (O(n^5) \cdot K) \cap (\mathbb{Z}^n \setminus \{0\}).$$

□

BIBLIOGRAPHY

[1] D. Aharonov and O. Regev. Lattice problems in NP cap conp. *J. ACM*, 52(5):749–765, 2005.

[2] M. Ajtai. Generating hard instances of lattice problems. In *Proceedings of the 28th STOC*, pages 99–108. ACM, 1996.

[3] N. Alon and J. H. Spencer. *The probabilistic method*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., Hoboken, NJ, third edition, 2008. With an appendix on the life and work of Paul Erdős.

[4] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

[5] W. Banaszczyk. Balancing vectors and Gaussian measures of n-dimensional convex bodies. *Random Structures Algorithms*, 12(4):351–360, 1998.

[6] N. Bansal. Constructive algorithms for discrepancy minimization. In *FOCS*, pages 3–10, 2010.

[7] N. Bansal, D. Dadush, and S. Garg. An algorithm for komlós conjecture matching banaszczyk's bound. *CoRR*, abs/1605.02882, 2016.

[8] N. Bansal and J. Spencer. Deterministic discrepancy minimization. *Algorithmica*, 67(4):451–471, 2013.

[9] J. Beck. Roth's estimate of the discrepancy of integer sequences is nearly sharp. *Combinatorica*, 1(4):319–325, 1981.

[10] J. Beck and T. Fiala. "Integer-making" theorems. *Discrete Appl. Math.*, 3(1):1–8, 1981.

[11] R. Bhatia. *Matrix analysis*, volume 169 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1997.

[12] T. Bohman. A sum packing problem of erdös and the conway-guy sequence. *Proceedings of the AMS*, 124(12):3627–3636, 1996.

[13] G. Bohus. On the discrepancy of 3 permutations. *Random Structures Algorithms*, 1(2):215–220, 1990.

[14] B. Chazelle. *The discrepancy method - randomness and complexity*. Cambridge University Press, 2001.

[15] R. Eldan and M. Singh. Efficient algorithms for discrepancy minimization in convex sets. *CoRR*, abs/1409.2913, 2014.

[16] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, New York, 1979.

[17] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1997.

[18] A. Giannopoulos. On some vector balancing problems. *Studia Mathematica*, 122(3):225–234, 1997.

[19] E. D. Gluskin. Extremal properties of orthogonal parallelepipeds and their applications to the geometry of banach spaces. *Mathematics of the USSR-Sbornik*, 64(1):85, 1989.

[20] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2, pages 122–125. Springer, 2012.

[21] I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. pages 469–477, 2007.

[22] R. Hoberg, H. Ramadas, T. Rothvoß, and X. Yang. Number balancing is as hard as minkowski's theorem and shortest vector. *CoRR*, abs/1611.08757, 2016.

[23] F. John. Extremum problems with inequalities as subsidiary conditions. In *Studies and Essays Presented to R. Courant on his 60th Birthday, January 8, 1948*, pages 187–204. Interscience Publishers, Inc., New York, N. Y., 1948.

[24] N. Karmarkar and R. Karp. The differencing method of set partitioning. Technical report, CS Division, UC Berkeley, 1982. http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-83-113.pdf.

[25] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2004.

[26] A. Lenstra, H. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[27] V. Lev and R. Yuster. On the size of dissociated bases. *the electronic journal of combinatorics*, 18(1):P117, 2011.

[28] A. Levy, H. Ramadas, and T. Rothvoß. Deterministic discrepancy minimization via the multiplicative weight update method. *CoRR*, abs/1611.08752, 2016.

[29] L. Lovász. *An algorithmic theory of numbers, graphs and convexity*. SIAM, 1986.

[30] L. Lovász. *Geometric algorithms and algorithmic geometry*. American Mathematical Society, 1990.

[31] S. Lovett and R. Meka. Constructive discrepancy minimization by walking on the edges. In *FOCS*, pages 61–67, 2012.

[32] W. Lunnon. Integer sets with distinct subset-sums. *Mathematics of Computation*, 50(181):297–320, 1988.

[33] A. Marcus, D. A Spielman, and N. Srivastava. Interlacing Families II: Mixed Characteristic Polynomials and the Kadison-Singer Problem. *ArXiv e-prints*, June 2013.

[34] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[35] J. Matoušek. *Geometric discrepancy*, volume 18 of *Algorithms and Combinatorics*. Springer-Verlag, Berlin, 1999. An illustrated guide.

[36] J. Matoušek. *Lectures on Discrete Geometry*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[37] R. Meka. Discrepancy and beating the union bound (`https://windowsontheory.org/2014/02/07/discrepancy-and-beating-the-union-bound/`). 2014.

[38] S. Mertens. The easiest hard problem: Number partitioning. *Computational Complexity and Statistical Physics*, 125(2):125–139, 2006.

[39] D. Micciancio. *On the hardness of the shortest vector problem*. PhD thesis, Massachusetts Institute of Technology, 1998.

[40] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

[41] A. Newman, O. Neiman, and A. Nikolov. Beck's three permutations conjecture: A counterexample and some consequences. In *FOCS*, pages 253–262, 2012.

[42] P. Nguyen and B. Vallée. The lll algorithm. *Information Security and Cryptography*, 2010.

[43] A. Nikolov. The Komlos Conjecture Holds for Vector Colorings. *ArXiv e-prints*, January 2013.

[44] C. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences*, 48(3):498–532, 1994.

[45] T. Rothvoß. Constructive discrepancy minimization for convex sets. In *FOCS*, pages 140–145, 2014.

[46] C. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

[47] J. Spencer. Balancing games. *J. Comb. Theory, Ser. B*, 23(1):68–74, 1977.

[48] J. Spencer. Six standard deviations suffice. *Transactions of the American Mathematical Society*, 289(2):679–706, 1985.

[49] J. H. Spencer, A. Srinivasan, and P. Tetali. The discrepancy of permutation families. Unpublished manuscript.

[50] A. Srinivasan. Improving the discrepancy bound for sparse matrices: Better approximations for sparse lattice approximation problems. In *SODA'97*, pages 692–701, Philadelphia, PA, 1997. ACM SIGACT, SIAM.

[51] G. Woeginger and Z. Yu. On the equal-subset-sum problem. *Information Processing Letters*, 42(6):299–302, 1992.

[52] A. Zouzias. A matrix hyperbolic cosine algorithm and applications. In *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, pages 846–858, 2012.