Approximating the Cut-Norm via Grothendieck's Inequality

Draft

Noga Alon * Assaf Naor †

Abstract

The *cut-norm* $||A||_C$ of a real matrix $A = (a_{ij})_{i \in R, j \in S}$ is the maximum, over all $I \subset R,\ J \subset S$ of the quantity $|\sum_{i \in I, j \in J} a_{ij}|$. This concept plays a major role in the design of efficient approximation algorithms for dense graph and matrix problems. Here we show that the problem of approximating the cut-norm of a given real matrix is MAX SNP hard, and provide an efficient approximation algorithm. This algorithm finds, for a given matrix $A = (a_{ij})_{i \in R, j \in S}$, two subsets $I \subset R$ and $J \subset S$, such that $|\sum_{i \in I, j \in J} a_{ij}| \geq \rho ||A||_C$, where $\rho > 0$ is an absolute constant satisfying $\rho > 0.56$. The algorithm combines semidefinite programming with a novel rounding technique based on Grothendieck's Inequality.

1 Introduction

The *cut-norm* $||A||_C$ of a real matrix $A = (a_{ij})_{i \in R, j \in S}$ with a set of rows indexed by R and a set of columns indexed by S is the maximum, over all $I \subset R,\ J \subset S$, of the quantity $|\sum_{i \in I, j \in J} a_{ij}|$. This concept plays a major role in the work of Frieze and Kannan on efficient approximation algorithms for dense graph and matrix problems, [7] (see also [2] and its references). Although the techniques in [7] enable the authors to approximate efficiently the cut-norm of an n by m matrix with entries in $[-1, 1]$ up to an **additive** error of ϵnm, there is no known polynomial algorithm that approximates the cut-norm of a general real matrix up to a constant multiplicative factor.

Let CUT NORM denote the computational problem of computing the cut-norm of a given real matrix. Here we first observe that the CUT NORM problem is MAX SNP hard, and then provide an efficient approximation algorithm for the problem. This algorithm finds, for a given matrix $A = (a_{ij})_{i \in R, j \in S}$, two subsets $I \subset R$ and $J \subset S$, such that $|\sum_{i \in I, j \in J} a_{ij}| \geq \rho ||A||_C$, where $\rho > 0$ is an absolute constant. We first describe a deterministic algorithm that supplies a rather poor value of ρ, and then describe a randomized algorithm that provides a solution of expected value greater than 0.56 times the optimum.

*Schools of Mathematics and Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel. Email: noga@math.tau.ac.il. Research supported in part by a USA-Israeli BSF grant, by the Israel Science Foundation and by the Hermann Minkowski Minerva Center for Geometry at Tel Aviv University.

†Microsoft Research, One Microsoft Way 113/2131, Redmond, WA 98052-6399, USA. Email: anaor@microsoft.com.

The algorithm combines semidefinite programming with a novel rounding technique based on (the proofs of) Grothendieck's Inequality. This inequality, first proved in [10], is a fundamental tool in Functional Analysis, and has several interesting applications in this area. We will actually use the matrix version of Grothendieck's inequality, formulated in [19]. In order to apply semidefinite programming for studying the cut-norm of an n by m matrix $A = (a_{ij})$, it is convenient to first study another norm,

$$||A||_{\infty \mapsto 1} = \max \sum_{i=1}^{n} \sum_{j=1}^{m} a_{ij} x_i y_j,$$

where the maximum is taken over all $x_i, y_j \in \{-1, 1\}$.

It is not difficult to show (see section 3), that for every matrix A,

$$4||A||_C \geq ||A||_{\infty \mapsto 1} \geq ||A||_C,$$

and hence a constant approximation of any of these norms provides a constant approximation of the other.

The value of $||A||_{\infty \mapsto 1}$ is given by the following quadratic integer program

$$\text{Maximize} \quad \sum_{ij} a_{ij} x_i y_j \tag{1}$$

$$\text{subject to} \quad x_i, y_j \in \{-1, 1\} \quad \text{for all} \quad i, j.$$

The obvious semidefinite relaxation of this program is

$$\text{Maximize} \quad \sum_{ij} a_{ij} u_i \cdot v_j \tag{2}$$

$$\text{subject to} \quad ||u_i|| = ||v_j|| = 1,$$

where here $u_i \cdot v_j$ denotes the inner product of u_i and v_j, which are now vectors of (Euclidean) norm 1 that lie in an arbitrary Hilbert space. Clearly we may assume, without loss of generality, that they lie in an $n + m$-dimensional space.

This semidefinite program can be solved, using well known techniques (see [9]) within an additive error of ϵ, in polynomial time (in the length of the input and in the logarithm of $1/\epsilon$.) The main problem is the task of rounding this solution into an integral one. A first possible attempt is to imitate the technique of Goemans and Williamson in [11], that is, given a solution u_i, v_j to the above program, pick a random vector z and define $x_i = \mathrm{sign(u_i \cdot z)}$ and $y_j = \mathrm{sign(v_j \cdot z)}$. It is easy to check that the expected value of $x_i y_j$ satisfies $E(x_i y_j) = \frac{2}{\pi} \arcsin(u_i \cdot v_j)$, and as $\arcsin(t)$ and t differ only in constant factors for all $-1 \leq t \leq 1$, one could hope that this will provide an integral solution whose value is at least some absolute constant fraction of the value of the optimal solution. This reasoning is, unfortunately, incorrect, as some of the entries a_{ij} may be positive and some may be negative, (in fact, the problem is interesting only if this is the case, since otherwise either $x_i = y_j = 1$ or $x_i = -y_j = 1$ for all i, j supplies the required maximum). Therefore, even if each single term $a_{ij} u_i \cdot v_j$ is approximated well by its integral rounding $a_{ij} x_i y_j$, there is no reason to expect the sum to be well-approximated, due to cancellations. We thus have to compare the value of the rounded solution to that of the semidefinite program on a global basis. Nesterov [21] obtained a result of this form for the problem of approximating the maximum value of a quadratic form $\sum_{ij} b_{ij} x_i x_j$, where $x_i \in \{-1, 1\}$, but only for the special case in which the matrix $B = (b_{ij})$ is positive semidefinite. While his estimate is global, his rounding is the same simple rounding technique of [11] described above. As explained before, some new ideas are required in our case in order to get any nontrivial result.

Luckily, there is a well known inequality of Grothendieck, which asserts that the value of the semidefinite program (2) and that of the integer program (1) can differ only by a constant factor. The precise value of this constant, called *Grothendieck's constant* and denoted by K_G, is not known, but it is known that its value is at most $\frac{\pi}{2\ln(1+\sqrt{2})} = 1.782...$ ([17]) and at least $\frac{\pi}{2} = 1.570...$ ([10]). Stated in other words, the integrability gap of the problem is at most K_G. (Krivine mentions in [17] that he can improve the lower bound, but such an improvement has never been published).

It follows that the value of the semidefinite program (2) provides an approximation of $||A||_{\infty \mapsto 1}$ up to a constant factor. This, however, still does not tell us how to round the solution of the semidefinite program into an integral one with a comparable value. Indeed, this task requires more work, and is carried out in the following sections.

We describe three rounding techniques. The first one is a deterministic procedure, which combines Grothendieck's Inequality with some facts about four-wise independent random variables, in a manner that resembles the technique used in [4] to approximate the second frequency moment of a stream of data under severe space constraints. The second rounding method is based on Rietz' proof of Grothendieck's Inequality [23]. This proof supplies a better approximation guarantee for the special case of positive semidefinite matrices A, where the integrality gap can be shown to be precisely $\pi/2$, and implies that Nesterov's analysis for the problem he considers in [21] is tight.

The third technique, which supplies the best approximation guarantee, is based on Krivine's proof of Grothendieck's Inequality. Here we use the vectors u_i, v_j which form a solution of the semidefinite program (2) to construct some other unit vectors u_i', v_j', which are first shown to exist in an infinite dimensional Hilbert space, and are then found, using another instance of semidefinite programming, in an $n+m$-dimensional space. These vectors can now be rounded to $\{-1, 1\}$ in order to provide an integral solution for the original problem (1) in a rather simple way. We note that there are several known techniques for modifying the solution of a semidefinite program before rounding it, see [25], [18], [8]. Here, however, the modification seems more substantial.

We believe that our techniques will have further applications, as they provide a method for handling problems in which there is a possible cancellation between positive and negative terms. It seems that there are additional interesting problems of this type. Moreover, unlike the semidefinite based approximation algorithms for MAX CUT, MAX 2SAT and related problems, suggested in the seminal paper of [11] and further developed in many subsequent papers, the problem considered here has no known constant approximation algorithm, and the semidefinite programming and its rounding appear to be essential in order to obtain any constant approximation guarantee, and not only in order to improve the constants ensured by appropriate combinatorial algorithms.

The rest of this paper is organized as follows. In Section 2 we present the (relatively simple) proof that the problem of approximating the cut-norm $||A||_C$, as well as the related problem of approximating $||A||_{\infty \mapsto 1}$, are both MAX SNP hard. In Section 3 we describe a deterministic procedure that approximates the cut-norm of a given matrix up to a constant factor. Two other methods, providing better constants, are described in Section 4, where we also consider the special case of positive semidefinite matrices. We conclude with Section 5, that includes some applications, concluding remarks and open problems.

2 Hardness of approximation

As usual, we say that an approximation algorithm for a maximization problem has *performance ratio* or *performance guarantee* ρ for some real $\rho \leq 1$, if it always delivers a solution with objective function value at least ρ times the optimum value. Such an approximation algorithm is then called a *ρ-approximation algorithm*. Similarly, a randomized approximation algorithm is a $\rho-$approximation algorithm if it always produces a solution with expected value at least ρ times the optimum.

In this section we observe that the problem of approximating the cut-norm $||A||_C$ of a given input matrix is MAX SNP hard, and so is the related problem of approximating $||A||_{\infty \mapsto 1}$. This implies, by the results in [22], [6], that there exists some $\rho < 1$ such that the existence of a ρ-approximation, polynomial-time algorithm for any of these problems would imply that $P = NP$. The proof is by a reduction of the MAX CUT problem to the CUT NORM problem, and to the problem of computing $||A||_{\infty \mapsto 1}$. We need the following simple observation.

Lemma 2.1 *For any real matrix $A = (a_{ij})$,*

$$||A||_C \leq ||A||_{\infty \mapsto 1} \leq 4||A||_C.$$

Moreover, if the sum of each row and the sum of each column of A is zero, then $||A||_{\infty \mapsto 1} = 4||A||_C$.

Proof: For any $x_i, y_j \in \{-1, 1\}$,

$$\sum_{i,j} a_{ij} x_i y_j = \sum_{i:x_i=1, j:x_j=1} a_{ij} - \sum_{i:x_i=1, j:x_j=-1} a_{ij} - \sum_{i:x_i=-1, j:x_j=1} a_{ij} + \sum_{i:x_i=-1, j:x_j=-1} a_{ij}.$$

The absolute value of each of the four terms in the right hand side is at most $||A||_C$, implying, by the triangle inequality, that

$$||A||_{\infty \mapsto 1} \leq 4||A||_C. \tag{3}$$

Suppose, now, that $||A||_C = \sum_{i \in I, j \in J} a_{ij}$ (the computation in case it is $-\sum_{i \in I, j \in J} a_{ij}$ is essentially the same). Define $x_i = 1$ for $i \in I$ and $x_i = -1$ otherwise, and similarly, $y_j = 1$ if $j \in J$ and $y_j = -1$ otherwise. Then

$$||A||_C = \sum_{i,j} a_{ij} \frac{1 + x_i}{2} \frac{1 + y_j}{2} = \frac{1}{4} \sum_{i,j} a_{ij} + \frac{1}{4} \sum_{i,j} a_{ij} x_i \cdot 1 + \frac{1}{4} \sum_{i,j} a_{ij} 1 \cdot y_j + \frac{1}{4} \sum_{i,j} a_{ij} x_i y_j.$$

The absolute value of each of the four terms in the right hand side is at most $||A||_{\infty\mapsto 1}/4$, implying that $||A||_{\infty\mapsto 1} \geq ||A||_C$. If the sum of each row and the sum of each column of A is zero, then the right hand side is precisely $\frac{1}{4}\sum_{i,j} a_{ij}x_iy_j$, implying that in this case $|A||_{\infty\mapsto 1} \geq 4||A||_C$, which, in view of (3), shows that the above holds as an equality. $\Box$

Proposition 2.2 *Given a (weighted or unweighted) graph $G = (V, E)$, there is an efficient way to construct a real $2|E|$ by $|V|$ matrix A, such that*

$$MAXCUT(G) = ||A||_C = ||A||_{\infty\mapsto 1}/4.$$

Therefore, the CUT NORM problem and the problem of computing $||A||_{\infty\mapsto 1}$ are both MAX SNP hard.

Proof: We describe the construction for the unweighted case. The weighted case is similar. Given $G = (V, E)$, orient it in an arbitrary manner. Let $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$, and let $A = (a_{ij})$ be the $2m$ by n matrix defined as follows. For each $1 \leq i \leq m$, if e_i is oriented from v_j to v_k, then $a_{2i-1,j} = a_{2i,k} = 1$ and $a_{2i-1,k} = a_{2i,j} = -1$. The rest of the entries of A are all 0. It is not difficult to check that $MAXCUT(G) = ||A||_C$. In addition, since the sum of entries in each row and in each column of A is zero, it follows by Lemma 2.1 that $||A||_{\infty\mapsto 1} = 4||A||_C$. As it is known [22], [13] that the MAX CUT problem is MAX SNP hard, the desired result follows. $\Box$

Håstad [13] has shown that if $P \neq NP$ then there is no polynomial time approximation algorithm for the MAX CUT problem with approximation ratio exceeding 16/17. Thus, this is an upper bound for the best possible approximation guarantee of a polynomial algorithm for approximating $||A||_C$ or $||A||_{\infty\mapsto 1}$. Similarly, our reduction above can be easily modified to construct, for any given directed graph D, a matrix B with vanishing row sums and column sums, so that the value of the maximum directed cut of D is equal to $||B||_C$. Håstad [13] has shown that if $P \neq NP$ then there is no polynomial time approximation algorithm for the MAX DICUT problem with approximation ratio exceeding 12/13. Thus, this is an upper bound for the best possible approximation guarantee of a polynomial approximation algorithm for $||B||_C$ or $||B||_{\infty\mapsto 1}$.

3 Approximating the cut-norm

In this section we describe an efficient, deterministic, ρ- approximation algorithm for the CUT NORM problem, where $\rho > 0$ is an absolute constant. We make no attempt here to optimize the value of ρ: this will be done (in a different way) in Section 4. We believe, however, that although the value of ρ obtained in this section is rather poor, the method, which is motivated by the proof Grothendieck's inequality in [15], is interesting and may lead to similar results for related problems.

Given a real n by m matrix $A = (a_{ij})$, our objective is to find $x_i, y_j \in \{-1, 1\}$, such that

$$\sum_{i,j} a_{ij}x_iy_j \geq \rho||A||_{\infty\mapsto 1},$$

where ρ is an absolute positive constant. The discussion in Section 1 and the proof of Lemma 2.1 imply that this will yield a similar procedure for finding I and J such that $|\sum_{i\in I,j\in J} a_{ij}| \geq \rho' ||A||_C$.

We start by solving the semidefinite program (2). We can thus compute, for any positive δ, unit vectors $u_i, v_j \in R^p$, where $p = n+m$, such that the sum $\sum_{i,j} a_{ij} u_i \cdot v_j$ is at least the maximum value of the program (2) (which is clearly at least $||A||_{\infty\mapsto 1}$) minus δ. Since the value of the above norm of A is at least the maximum absolute value of an entry of A, we can make sure that the δ term is negligible. The main part of the algorithm is the rounding phase, that is, the phase of finding, using the vectors u_i, v_j, reals $x_i, y_j \in \{-1, 1\}$, such that

$$\sum_{i,j} a_{ij} x_i y_j \geq \rho \sum_{i,j} a_{ij} u_i \cdot v_j.$$

This is done as follows. Let V be an explicit set of $t = O(p^2)$ vectors $\epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_p) \in \{-1,1\}^p$ in which the values ϵ_j are four-wise independent and each of them attains the two values -1 and 1 with equal probability. This means that for every four distinct coordinates $1 \leq i_1 < \ldots < i_4 \leq n$ and every choice of $\epsilon_1, \ldots, \epsilon_4 \in \{-1,1\}$, exactly a $(1/16)-$fraction of the vectors have ϵ_j in their coordinate number i_j for $j = 1, \ldots, 4$. As described, for example, in [1] or [5] such sets (also known as *orthogonal arrays of strength* 4) can be constructed efficiently using the parity check matrices of BCH codes.

Let $M > 0$ be a fixed real, to be chosen later. Consider V as a sample space in which all t points ϵ have the same probability. For any unit vector $q = (q_1, q_2 \ldots, q_p) \in R^p$, let $H(q)$ denote the random variable defined on the sample space V by putting $[H(q)](\epsilon) = \sum_{j=1}^p \epsilon_j q_j$. Since the entries ϵ_j are four-wise (and hence pairwise) independent, it follows that for any two vectors q, q', the expectation of $H(q)H(q')$ is precisely the inner product $q \cdot q'$. In particular, the expectation of $[H(q)]^2$ is $q \cdot q = 1$. Similarly, four-wise independence implies that the expectation of $[H(q)]^4$ satisfies

$$E([H(q)]^4) = \sum_{j=1}^p q_j^4 + 6 \sum_{1\leq j<j'\leq p} q_j^2 q_{j'}^2 \leq 3\left(\sum_j q_j^2\right)^2 = 3.$$

The *M-truncation* of $H(q)$, denoted $H^M(q)$, is defined as follows: $[H^M(q)](\epsilon) = [H(q)](\epsilon)$ if $|[H(q)](\epsilon)| \leq M$, $[H^M(q)](\epsilon) = M$ if $[H(q)](\epsilon) > M$, and $[H^M(q)](\epsilon) = -M$ if $[H(q)](\epsilon) < -M$.

By Markov's Inequality, for every positive real m, $Prob[\ |H(q)| \geq m] m^4 \leq E([H(q)]^4) \leq 3$, implying that $Prob[\ |H(q)| \geq m] \leq \frac{3}{m^4}$. This implies the following

Claim 3.1 *The expectation* $E(|H(q) - H^M(q)|^2)$ *satisfies*

$$E(|H(q) - H^M(q)|^2) \leq \frac{1}{M^2}.$$

□

Each random variable $H(q)$ can be associated with a vector $h(q) \in R^t$ by defining $[h(q)](\epsilon) = \frac{1}{\sqrt{t}}[H(q)](\epsilon)$. The truncation $h^M(q) = H^M(q)/\sqrt{t}$ is defined analogously. The above discussion thus implies the following.

Lemma 3.2 *For each unit vector $q \in R^p$, $h(q) \in R^t$ is a unit vector. The norm of $h^M(q)$ is at most 1, and that of $h(q) - h^M(q)$ is at most $1/M$. If $q' \in R^p$ is another vector, then $h(q) \cdot h(q') = q \cdot q'$.* $\square$

Returning to our semidefinite program (2) and its solution (up to δ) given by the vectors $u_i, v_j \in R^p$, let B denote the value of the program. Since $h(q) \cdot h(q') = q \cdot q'$ for all unit vectors q, q', it follows that

$$B - \delta \leq \sum_{ij} a_{ij} u_i \cdot v_j = \sum_{ij} a_{ij} h(u_i) \cdot h(v_j)$$

$$= \sum_{ij} a_{ij} h^M(u_i) \cdot h^M(v_j) + \sum_{ij} a_{ij}(h(u_i) - h^M(u_i)) \cdot h^M(v_j) + \sum_{ij} a_{ij} h(u_i) \cdot (h(v_j) - h^M(v_j)).$$

By the convexity of the program, and since the norm of each vector $h^M(v_j), h(u_i)$ is at most 1, and the norm of each vector $h(u_i) - h^M(u_i)$ and each vector $h(v_j) - h^M(v_j)$ is at most $1/M$, it follows that

$$\sum_{ij} a_{ij}(h(u_i) - h^M(u_i)) \cdot h^M(v_j) + \sum_{ij} a_{ij} h(u_i) \cdot (h(v_j) - h^M(v_j)) \leq \frac{2}{M} B.$$

Here we have used, crucially, the fact that as B is the maximum value of the semidefinite program, its value on the vectors $h(u_i) - h^M(u_i)$ and the vectors $h(v_j)$ does not exceed $B\frac{1}{M}$, and so does its value on the vectors $h(u_i)$ and $h(v_j) - h^M(v_j)$. Therefore,

$$B(1 - \frac{2}{M}) - \delta \leq \sum_{ij} a_{ij} h^M(u_i) \cdot h^M(v_j).$$

It follows that there is a coordinate $\epsilon \in V$ such that

$$\sum_{ij} a_{ij} h^M(u_i)(\epsilon) \cdot h^M(v_j)(\epsilon) \geq \frac{1}{t}\left(B\left(1 - \frac{2}{M}\right) - \delta\right).$$

By the definition of the vectors h, this implies

$$\sum_{ij} a_{ij} H^M(u_i)(\epsilon) \cdot H^M(v_j)(\epsilon) \geq B\left(1 - \frac{2}{M}\right) - \delta.$$

Choose $M = 3$ and define $x_i = H^M(u_i)(\epsilon)/M$, $y_j = H^M(v_j)(\epsilon)/M$. Then x_i, y_j are reals, each having an absolute value at most 1, and

$$\sum_{ij} a_{ij} x_i y_j \geq B\left(\frac{M-2}{M^3}\right) - \frac{\delta}{M^2} = \frac{B}{27} - \frac{\delta}{9}.$$

Fixing all x_i, y_j but, say, x_1, the left hand side is a linear form in x_1, and thus we can shift x_1 to either -1 or 1, without any decrease in the value of the sum. Proceeding in this way with the other variables, each one in its turn, we obtain $x_i, y_j \in \{-1, 1\}$ such that the value of the sum $\sum_{ij} a_{ij} x_i y_j$ is at least $\frac{B}{27} - \frac{\delta}{9}$. As δ is arbitrarily small, we have thus proved the following.

Theorem 3.3 *There is a deterministic polynomial time algorithm that finds, for a given real matrix $A = (a_{ij})$, integers $x_i, y_j \in \{-1, 1\}$ such that the value of the sum $\sum_{ij} a_{ij} x_i y_j$ is at least* $0.03\ B$, *where B is the value of the semidefinite program (2) (which is at least $||A||_{\infty \mapsto 1}$).* $\square$

4 Improving the constant

The constant obtained in the previous section can be improved by replacing the space V of four-wise independent $\{-1, 1\}$ variables with a space of $2k$-wise independent $\{-1, 1\}$ variables (or with a space of independent standard normal random variables). This, however, will not provide a ρ-approximation algorithm with $\rho > \frac{1}{5}$.

In fact, we can do much better. In this section we describe two randomized ρ-approximation algorithms for approximating $||A||_{\infty \mapsto 1}$. For the first algorithm, $\rho = \frac{4}{\pi} - 1 > 0.27$, while for the second $\rho = \frac{2\ln(1+\sqrt{2})}{\pi} > 0.56$. We further show that these yield randomized ρ-approximation algorithms for the CUT NORM problem, with the same values of ρ (without losing the factor of 4 that appears in Lemma 2.1.) It should be possible to derandomize these algorithms using the techniques of [20].

4.1 Averaging with a Gaussian measure

The main idea here, based on [23], is to round the solution $u_i, v_j \in R^p$ of the semidefinite program (2) by averaging over R^p with normalized Gaussian measure. We proceed with the details.

Let $g_1, g_2, \ldots, g_p$ be standard, independent, Gaussian random variables, and consider the random vector $G = (g_1, \ldots, g_p)$. The following identity holds for every two unit vectors $b, c \in \ell_2^p$:

$$\frac{\pi}{2} E\left[\mathrm{sign}(\mathrm{b} \cdot \mathrm{G}) \cdot \mathrm{sign}(\mathrm{c} \cdot \mathrm{G})\right] = b \cdot c + E\left\{\left[b \cdot G - \sqrt{\frac{\pi}{2}}\mathrm{sign}(\mathrm{b} \cdot \mathrm{G})\right] \cdot \left[c \cdot G - \sqrt{\frac{\pi}{2}}\mathrm{sign}(\mathrm{c} \cdot \mathrm{G})\right]\right\} \quad (4)$$

This is a simple exercise, using rotation invariance. Indeed, the fact that

$$E\left[(b \cdot G)(c \cdot G)\right] = b \cdot c \quad (5)$$

follows from the orthogonality of the vectors g_i; if $b = (b_1, \ldots, b_p)$ and $c = (c_1, \ldots, c_p)$, then

$$E\left[(b \cdot G)(c \cdot G)\right] = E\left[\sum_{i=1}^{p} b_i g_i \sum_{i=1}^{p} c_i g_i\right] = \sum_{i,j} b_i c_j E[g_i g_j] = \sum_{i=1}^{p} b_i c_i = b \cdot c.$$

To compute $E\left[(b \cdot G)\mathrm{sign}(\mathrm{c} \cdot \mathrm{G})\right]$ assume, by rotation invariance, that $c = (1, 0, \ldots, 0)$, and $b = (b_1, b_2, 0, \ldots, 0)$. Hence $b \cdot c = b_1$ and

$$E\left[(b \cdot G)\mathrm{sign}(\mathrm{c} \cdot \mathrm{G})\right] = E\left[(b_1 g_1 + b_2 g_2)\mathrm{sign}(\mathrm{g}_1)\right] = E\left[(b_1 g_1 \mathrm{sign}(\mathrm{g}_1)\right] + E[b_2 g_2]E[\mathrm{sign}(\mathrm{g}_1)]$$

$$= E\left[(b_1 g_1 \mathrm{sign}(\mathrm{g}_1)\right] = 2\int_0^{\infty} \frac{1}{\sqrt{2\pi}} b_1 x e^{-x^2/2} dx = \sqrt{\frac{2}{\pi}} b_1.$$

Thus

$$E\left[(b \cdot G)\mathrm{sign}(\mathrm{c} \cdot \mathrm{G})\right] = \sqrt{\frac{2}{\pi}}\ b \cdot c. \quad (6)$$

The identity above follows from (5) and (6), by linearity of expectation.

Suppose, now, that the vectors $u_i, v_j \in R^p$ supply a solution of (2), which can be found efficiently (we assume, for simplicity, that this is a precise solution). Let B denote the value of the solution. By (4)

$$\frac{\pi}{2} E \left\{ \sum_{i,j} a_{ij} \left[\mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G}) \cdot \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{G}) \right] \right\}$$

$$= B + \sum_{i,j} a_{ij} E \left\{ \left[u_i \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G}) \right] \cdot \left[v_j \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{G}) \right] \right\}. \tag{7}$$

Note that each term of the form

$$E \left\{ \left[u_i \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G}) \right] \cdot \left[v_j \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{G}) \right] \right\}$$

which multiplies a_{ij} in (7), is the inner product of two vectors in a Hilbert space. Moreover, the square of the norm of each of these vectors is easily seen to be $\frac{\pi}{2} - 1$, by substituting $b = c = u_i$ (or $b = c = v_j$) in (4). Therefore, as B is the maximum possible value of the program (2), it follows that the last sum in (7) is bounded, in absolute value, by $(\frac{\pi}{2} - 1)B$. Thus, by (7)

$$\frac{\pi}{2} E \left\{ \sum_{i,j} a_{ij} \left[\mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G}) \cdot \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{G}) \right] \right\} \geq \left(2 - \frac{\pi}{2} \right) B,$$

implying that by choosing the normal random variables g_i randomly, and by defining

$$x_i = \mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G}), \;\; \mathrm{y_j} = \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{G})$$

we get a solution for (1) whose expected value is at least $\frac{2}{\pi}(2 - \frac{\pi}{2})B = (\frac{4}{\pi} - 1)B$. As B is at least the value of the optimal solution of (1), this supplies a randomized ρ-approximation algorithm for estimating $||A||_{\infty \mapsto 1}$, where $\rho = \frac{4}{\pi} - 1$.

Remark: The above algorithm is based only on the basic idea in the proof of [23], and it is in fact possible to improve its performance guarantee by modifying it according to the full proof. This suggests to round $u_i \cdot G$ to $x_i = \mathrm{sign}(\mathrm{u_i} \cdot \mathrm{G})$ if it is "large", and to round it to some multiple of $u_i \cdot G$ if it is not, where the definition of "large" and the precise multiple are chosen optimally. We can then further round the fractional values of x_i to integral ones with no loss. This resembles some of the ideas used in several recent papers including [25], [18] and [8]. We do not include the details here, as we can get a better approximation guarantee, using another method, described in subsection 4.3.

4.2 Positive semidefinite matrices

In this section we observe that when $A = (a_{ij})$ is positive semidefinite then the approximation ratio can be improved to $2/\pi$. This follows from the work of Nesterov [21], but the short argument we describe here is based on Rietz' proof that for such an A the constant in Grothendieck's Inequality can be improved to $\pi/2$. Grothendieck himself showed in [10] (in a somewhat different language) that $\pi/2$ is a lower bound for the constant in this case; we sketch a proof of this fact below.

We first show that if $A = (a_{ij})$ is a positive semidefinite n by n matrix, then the maximum of the semidefinite program (2) is obtained for some vectors u_i, v_j satisfying $u_i = v_i$ for all i (though, of course, it may also be obtained for some vectors that do not satisfy this property). This is not really required for getting the $2/\pi$ approximation algorithm, but we include this proof as it shows that the integrality gap of the problem is at most $\pi/2$.

Suppose, thus, that the maximum value of (2), denoted by B, is obtained by some p-dimensional vectors, where $p \leq 2n$ and A is positive semidefinite. Let $D = A \otimes I$ be the tensor product of A with a p by p identity matrix. This is simply the np by np matrix consisting of p blocks along the diagonal, each being a copy of A. Given $u_1, \ldots, u_n \in \ell_2^p$, where $u_i = (u_{i1}, u_{i2}, \ldots, u_{ip})$ let $u^{(j)} = (u_{1j}, u_{2j}, \ldots, u_{nj})$ be the vector consisting of the j-th coordinates of all vectors u_i, $(1 \leq j \leq p)$. Let $u = (u^{(1)}, u^{(2)}, \ldots, u^{(p)}) \in \ell_2^{np}$ and note that $Du = (Au^{(1)}, Au^{(2)}, \ldots, Au^{(p)})$. Thus D is positive semidefinite and for $u_1, \ldots, u_n, v_1, \ldots, v_n \in \ell_2^p$,

$$\sum_{i,j} a_{ij} u_i \cdot v_j = Du \cdot v = D^{1/2}u \cdot D^{1/2}v \leq \|D^{1/2}u\| \cdot \|D^{1/2}v\|,$$

with equality when $u = v$.

Therefore, if the maximum B of the quantity $\sum_{i,j} a_{ij} u_i \cdot v_j$ is obtained for the unit vectors $u_i, v_j \in \ell_2^p$, then, as $\|D^{1/2}u\|$ cannot exceed $B^{1/2}$, it follows that $\|D^{1/2}u\| = \|D^{1/2}v\| = B^{1/2}$. Thus the maximum is also equal to $\sum_{i,j} a_{ij} u_i \cdot u_j$ (and to $\sum_{i,j} a_{ij} v_i \cdot v_j$).

The fact that for positive semidefinite matrices A we can get a ρ-approximation algorithm with $\rho = \frac{2}{\pi}$ is now an easy consequence of (7). We first solve the semidefinite program (2) to get vectors u_i, v_j optimizing it. By the above discussion, the vectors $u_i = v_i$ for all i also give an optimal solution. (Alternatively, we can solve the variant of (2) in which $u_i = v_i$ for all i directly, and proceed from there.) By (7)

$$\frac{\pi}{2} E\left\{\sum_{i,j} a_{ij} \left[\mathrm{sign(u_i \cdot G) \cdot sign(u_j \cdot G)}\right]\right\}$$

$$= B + \sum_{i,j} a_{ij} E\left\{\left[u_i \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign(u_i \cdot G)}\right] \cdot \left[u_j \cdot G - \sqrt{\frac{\pi}{2}} \cdot \mathrm{sign(u_j \cdot G)}\right]\right\} \geq B,$$

where here we have used the fact that A is positive semidefinite. The algorithm now simply chooses G at random, and computes $x_i = y_i = \mathrm{sign(u_i \cdot G)}$.

We conclude this subsection with a sketch of (a modified version of) the argument of Grothendieck that shows that the integrality gap of (1) for the positive semidefinite case is indeed precisely $\pi/2$. We need the following

Fact: If b and c are two random, independent, vectors on the unit sphere in R^p, then

$$E[(b \cdot c)^2] = \frac{1}{p} \tag{8}$$

and

$$E[|b \cdot c|] = \left(\sqrt{\frac{2}{\pi}} + o(1)\right)\frac{1}{\sqrt{p}}, \tag{9}$$

where the $o(1)$-term tends to zero as p tends to infinity.

Indeed, the computation of the first expectation is very simple; by rotation invariance we may assume that $c = (1, 0, 0, \ldots, 0)$ and then $E[(b \cdot c)^2] = E[b_1^2]$, where $b = (b_1, b_2, \ldots, b_p)$. However, by symmetry $E[b_1^2] = \frac{1}{p}E[b_1^2 + b_2^2 + \ldots + b_p^2] = \frac{1}{p}$, implying (8). By the same reasoning, $E[|b \cdot c|] = E[|b_1|]$ which can now either be computed directly by integrating along the sphere, or can be estimated by noticing that it is well approximated by $E[|\frac{g}{\sqrt{p}}|]$, where g is a standard Gaussian random variable. The simple computation for this case appears before the derivation of (6).

Armed with the above fact, we now define an n by n positive semidefinite matrix $A = (a_{ij})$ for which the ratio between the value of (1) and that of (2) is (nearly) $\pi/2$. Fix a large integer p, and let n be much larger. Let $v_1, v_2, \ldots, v_n$ be n independent random vectors chosen uniformly according to the normalized Haar measure on the unit sphere in R^p. Let A be the gram matrix of the vectors $\frac{v_i}{n}$, that is $a_{ij} = \frac{1}{n^2} v_i \cdot v_j$. Obviously A is positive semidefinite. Moreover, if we substitute the unit vectors v_i in the program (2) we get

$$\sum_{i,j} a_{ij} v_i \cdot v_j = \frac{1}{n^2} \sum_{ij} (v_i \cdot v_j)^2.$$

When n tends to infinity, this converges to the average value of the square of the inner product between two random vectors on the unit sphere of R^p, which is, by (8), $1/p$. Therefore, the optimal value of the program (2) for A is at least $1/p$.

Consider, now, the optimal value of the integer program (1) for A. Let $x_i \in \{-1, 1\}$ be a sign vector. Then

$$\sum_{i,j} a_{ij} x_i \cdot x_j = \|\frac{1}{n} \sum_{i=1}^{n} x_i v_i\|^2.$$

Therefore, the value of the integer program is the square of the maximum possible norm of a vector $\frac{1}{n}\sum_{i=1}^n x_i v_i$, where $x_i \in \{-1, 1\}$ for all i. If the direction of this optimal vector is given by the unit vector c, then, knowing c, it is clear how to choose x_i for each i; it simply has to be $\mathrm{sign(v_i \cdot c)}$. With this choice of the signs x_i, the quantity

$$\frac{1}{n} \sum_{i=1}^{n} x_i v_i \cdot c = \|\frac{1}{n} \sum_{i=1}^{n} x_i v_i\|$$

converges, when n tends to infinity, to the average value of $|v \cdot c|$, where v is a random vector on the sphere. By (9) this value is $(\sqrt{2/\pi} + o(1))\frac{1}{\sqrt{p}}$, where the $o(1)$-term tends to zero as p tends to infinity. Since n can be chosen to be arbitrarily large with respect to p, and as we do not have to consider all the infinitely many possible directions c, but can consider an appropriate ϵ-net of directions on the sphere, we conclude that if p is large and n is huge, then, with high probability, the value of the integer program (1) for A is at most

$$\left[\left(\sqrt{\frac{2}{\pi}} + o(1)\right)\frac{1}{\sqrt{p}}\right]^2 = \left(\frac{2}{\pi} + o(1)\right)\frac{1}{p}.$$

It follows that the integrality gap is at least $\pi/2$, implying, by the discussion in the beginning of this subsection, that it is exactly $\pi/2$.

4.3 Constructing new vectors

Our approach in this subsection is based on a simplified version of Krivine's proof, presented in [17], of Grothendieck's Inequality, as described in [14]. We need the following simple lemma, which has already been mentioned briefly in the introduction, and which has been applied in Grothendieck's original proof as well.

Lemma 4.1 (Grothendieck's identity) *For every two unit vectors u, v in a Hilbert space, if z is chosen randomly and uniformly according to the normalized Haar measure on the unit sphere of the space, then*

$$\frac{\pi}{2} \cdot E([\mathrm{sign}\,(\mathrm{u} \cdot \mathrm{z})] \cdot [\mathrm{sign}\,(\mathrm{v} \cdot \mathrm{z})]) = \arcsin\,(u \cdot v)\,.$$

□

Using this lemma, we prove the following.

Lemma 4.2 *For any set $\{u_i : \ 1 \leq i \leq n\} \cup \{v_j : 1 \leq j \leq m\}$ of unit vectors in a Hilbert space H, and for $c = \sinh^{-1}(1) = \ln(1+\sqrt{2})$, there is a set $\{u'_i : \ 1 \leq i \leq n\} \cup \{v'_j : 1 \leq j \leq m\}$ of unit vectors in a Hilbert space H', such that if z is chosen randomly and uniformly in the unit sphere of H' then*

$$\frac{\pi}{2} \cdot E\left([\mathrm{sign}\,(\mathrm{u'_i} \cdot \mathrm{z})] \cdot \left[\mathrm{sign}\left(\mathrm{v'_j} \cdot \mathrm{z}\right)\right]\right) = c\ u_i \cdot v_j$$

for all $1 \leq i \leq n, \ 1 \leq j \leq m$.

Proof: Fix c as above, a Hilbert space H and $u, v \in H$. By Taylor's expansion:

$$\sin\,(c\ u \cdot v) = \sum_{k=0}^{\infty} (-1)^k \frac{c^{2k+1}}{(2k+1)!}\,(u \cdot v)^{2k+1}\,.$$

For every vector w and integer j denote by $w^{\otimes j}$ the j'th tensor power $w \otimes w \otimes \cdots \otimes w$ (j terms). Then the above expansion becomes

$$\sin\,(c\ u \cdot v) = \sum_{k=0}^{\infty} (-1)^k \frac{c^{2k+1}}{(2k+1)!} u^{\otimes(2k+1)} \cdot v^{\otimes(2k+1)}.$$

Consider the following vectors in the direct sum $\oplus_{k=0}^{\infty} H^{\otimes(2k+1)}$, whose k'th "coordinate" is given by:

$$T(u)_k = (-1)^k \sqrt{\frac{c^{2k+1}}{(2k+1)!}} \cdot u^{\otimes(2k+1)} \quad \text{and} \quad S(y)_k = \sqrt{\frac{c^{2k+1}}{(2k+1)!}} \cdot v^{\otimes(2k+1)}$$

Then the above expansion boils down to $\sin\,(c\ u \cdot v) = T(u) \cdot S(v)$, or

$$c\ u \cdot v = \arcsin\,(T(u) \cdot S(v))\,.$$

Moreover,

$$\|T(u)\|^2 = \sinh\left(c \cdot \|u\|^2\right) \quad \text{and} \quad \|\mathrm{S(v)}\|^2 = \sinh\left(\mathrm{c} \cdot \|\mathrm{v}\|^2\right).$$

Given the unit vectors u_i, v_j, recall that $c = sinh^{-1}(1)$ and define $u'_i = T(u_i)$ and $v'_j = S(v_i)$. Note that all the vectors u'_i, v'_j are unit vectors (in the huge direct sum of tensor products we constructed). Let H' be the span of u'_i, v'_j. It is an $m + n$-dimensional Hilbert space. Let z be a random vector chosen uniformly on its unit sphere. By Grothendieck's identity (Lemma 4.1), for every i, j:

$$\frac{\pi}{2} \cdot E([\mathrm{sign}\,(\mathrm{T}(\mathrm{u_i}) \cdot \mathrm{z})] \cdot [\mathrm{sign}\,(\mathrm{S}(\mathrm{v_j}) \cdot \mathrm{z})]) = \arcsin\,(T(u_i) \cdot S(v_j)) = c\ u_i \cdot v_j,$$

as needed. $\square$

Theorem 4.3 *There is a randomized polynomial time algorithm that given an input n by m matrix* $A = (a_{ij})$ *and unit vectors* u_i, v_j *in* R^{n+m} *finds* $x_i, y_j \in \{-1, 1\}$ *such that the expected value of the sum* $\sum_{ij} a_{ij} x_i y_j$ *is*

$$\frac{2\ln(1+\sqrt{2})}{\pi} \sum_{ij} a_{ij} u_i \cdot v_j.$$

Therefore, there is a polynomial randomized ρ*-approximation algorithm for computing* $||A||_{\infty \mapsto 1}$*, where* $\rho = \frac{2\ln(1+\sqrt{2})}{\pi} > 0.56$*.*

Proof: By Lemma 4.2 there are vectors u'_i, v'_j satisfying the conclusion of the lemma. We can thus find such vectors in an $m + n$-dimensional space, using semidefinite programming. By linearity of expectation, with $c = sinh^{-1}(1)$ as above,

$$c \cdot \sum_{i,j} a_{ij} u_i \cdot v_j = \frac{\pi}{2} \cdot E(\sum_{i,j} a_{ij}\,[\mathrm{sign}\,(\mathrm{u'_i} \cdot \mathrm{z})] \cdot \Big[\mathrm{sign}\,\Big(\mathrm{v'_j} \cdot \mathrm{z}\Big)\Big]).$$

We can now simply pick a random z and define $x_i = \mathrm{sign}(\mathrm{u'_i} \cdot \mathrm{z})$ and $y_j = \mathrm{sign}(\mathrm{v_j} \cdot \mathrm{z})$. $\square$

4.4 The cut-norm

In this short subsection we observe that the same approximation ratio guaranteed in any approximation algorithm for $||A||_{\infty \mapsto 1}$ can be obtained for the CUT NORM problem as well. Given an n by m matrix $A = a_{ij}$, augment it to an $(n+1)$ by $(m+1)$ matrix $A' = (a'_{ij})$ by defining $a'_{ij} = a_{ij}$ for all $1 \leq i \leq n,\ 1 \leq j \leq m$, $a'_{i,m+1} = -\sum_{j=1}^{m} a_{ij}$ for all $1 \leq i \leq n$, $a'_{n+1,j} = -\sum_{i=1}^{n} a_{ij}$ for all $1 \leq j \leq m$, and $a'_{n+1,m+1} = 0$. We claim that $||A'||_C = ||A||_C$. Indeed, obviously $||A'||_C \geq ||A||_C$, as A is a submatrix of A'. Conversely, let $I \subset \{1, 2, \ldots, n+1\},\ J \subset \{1, 2, , \ldots, m+1\}$ satisfy $||A'||_C = |\sum_{i \in I, j \in J} a'_{ij}|$. If $n+1 \in I$ replace it by its complement $\{1, 2, \ldots, n+1\} \setminus I$ and similarly, if $m+1 \in J$, replace it by its complement $\{1, 2, \ldots, m+1\} \setminus J$. As the sum of each row and the sum of each column of A' is zero, the absolute value of the sum $\sum_{i \in I, j \in J} a'_{ij}$ with the new sets I, J is still equal to $||A'||_C$. This is, however, the sum of elements of a submatrix of A, implying that in fact $||A'||_C = ||A||_C$, as claimed.

By the last part of Lemma 2.1, $||A'||_{\infty \mapsto 1} = 4||A'||_C$, and thus we can simply apply any algorithm for ρ-approximating $||A'||_{\infty \mapsto 1}$ to obtain a similar ρ-approximation of $||A'||_C = ||A||_C$.

5 Concluding remarks

- The Regularity Lemma of Szemerédi [24] is a result that asserts that any graph can be partitioned in a certain regular way. An algorithmic version of this lemma appears in [3], together with several algorithmic applications. The main step in the design of this algorithmic version, is an efficient procedure for approximating the "regularity" of a pair of subsets in a given graph. This step can be improved using our results here. To state this improvement, we need a few definitions.

  Let $G = (V, E)$ be an undirected graph, and let A, B be two disjoint nonempty subsets of V. Let $e(A, B)$ denote the number of edges of G with an endpoint in A and an endpoint in B, and define the *density of edges* between A and B by $d(A, B) = \frac{e(A,B)}{|A||B|}$. For $\epsilon > 0$, the pair (A, B) is called *ϵ-regular* if for every $X \subset A$ and $Y \subset B$ satisfying $|X| \geq \epsilon|A|$ and $|Y| \geq \epsilon|B|$, the inequality

  $$|d(A, B) - d(X, Y)| < \epsilon$$

  holds.

  The main step in [3] is a polynomial time algorithm that, given two disjoint subsets $A, B \subset V$ in a graph G, where $|A| = |B| = n$, and given $\epsilon > 0$, either decides that the pair (A, B) is ϵ-regular, or finds two subsets $X \subset A$ and $Y \subset B$, each of size at least $\frac{\epsilon^4}{16}n$, such that $|d(A, B) - d(X, Y)| \geq \epsilon^4$. A somewhat better result can be obtained using the approximation algorithms developed here. Given $G = (V, E)$, $\epsilon > 0$ and A, B as above, compute, first, the density $d = d(A, B)$. Let $F = (f_{ab})_{a\in A, b\in B}$ be the n by n matrix defined by $f_{ab} = 1 - d$ if $ab \in E$ and $f_{ab} = -d$ if $ab \notin E$. Note that if (A, B) is not ϵ-regular, then there are $I \subset A,\ J \subset B$ satisfying $|I| \geq \epsilon n,\ |J| \geq \epsilon n$ such that

  $$|\sum_{a\in I, b\in J} f_{ab}| \geq \epsilon|I||J| \geq \epsilon^3 n^2,$$

  that is, the cut-norm of F is at least $\epsilon^3 n^2$. Therefore, in this case our algorithm will find efficiently $X \subset A, Y \subset B$ such that

  $$|\sum_{a\in X, b\in Y} f_{ab}| \geq 0.56\epsilon^3 n^2.$$

  Obviously this implies, say, that $|X| \geq 0.5\epsilon^3 n,\ |Y| \geq 0.5\epsilon^3 n$ and $|d(X, Y) - d(A, B)| \geq 0.5\epsilon^3$. If the algorithm does not find such sets, it can report that the pair (A, B) is ϵ-regular. This can be used instead of Corollary 3.3 in [3] to obtain efficiently a regular partition of any given graph with less parts than the ones ensured by the algorithm in [3] (but the number will still be huge; a tower of height polynomial in $1/\epsilon$, the degree of this polynomial will be smaller than the one in [3]).

- In [7] Frieze and Kannan describe an efficient algorithm for finding what they call a *cut-decomposition* of a given n by m real matrix A, and apply it to obtain efficient approximation algorithms for various dense graph and matrix problems.

Consider matrices with a set of rows indexed by R and a set of columns indexed by S. For $I \subset R$ and $J \subset S$, and for a real d, the *cut matrix* $D = CUT(I, J, d)$ is the matrix $(d_{ij})_{i \in R, j \in S}$ defined by $d_{ij} = d$ if $i \in I, j \in J$ and $d_{ij} = 0$ otherwise. A *cut-decomposition* of A expresses it in the form

$$A = D^{(1)} + \ldots + D^{(s)} + W,$$

where the matrices $D^{(i)}$ are cut matrices, and the matrix $W = (w_{kl})$ has a relatively small cut-norm.

The authors of [7] describe an efficient algorithm that produces, for any given n by m matrix A with entries in $[-1, 1]$, a cut-decomposition in which the number of cut matrices is $O(1/\epsilon^2)$, and the cut-norm of the matrix W is at most ϵnm. This is done as follows. Starting with $W^{(0)} = A$, suppose that the first i cut matrices $D^{(1)}, \ldots, D^{(i)}$ have already been defined, and consider the difference $W^{(i)} = A - \sum_{j=1}^{i} D^{(j)}$. If the cut-norm of this difference is already smaller than ϵnm, we are done. Otherwise, let I, J be sets of rows and columns such that

$$|\sum_{k \in I, l \in J} w_{kl}^{(i)}| \geq \rho \epsilon nm, \tag{10}$$

where $\rho > 0$ is an absolute positive constant. Let d be the average value of the entries $w_{kl}^{(i)}$ for $k \in I, l \in J$, and define $D^{(i+1)} = CUT(I, J, d), W^{(i+1)} = W^{(i)} - D^{(i+1)}$. A simple computation, described in [7], shows that the sum of squares of the entries of the new matrix $W^{(i+1)}$ is at most the sum of squares of the entries of the matrix $W^{(i)}$ minus $\rho^2 \epsilon^2 nm$. Therefore, this process must terminate after at most $\frac{1}{\rho^2 \epsilon^2}$ steps. The main step in the algorithm is clearly that of finding the sets I, J that satisfy (10), and the authors are able to do it efficiently only when the cut-norm is at least a constant fraction of nm. Our algorithm here enables us to perform this task efficiently (even if ϵ is, say, $1/n^{0.001}$, which is not feasible using the approach of [7], as the running time of their algorithm is exponential in $1/\epsilon^2$).

- The rounding techniques described in Section 3 or in Subsection 4.1 (but not the one described in Subsection 4.3) can be used to find efficiently, for any given square n by n matrix $A = (a_{ij})$, a vector $(x_1, \ldots, x_n) \in \{-1, 1\}$, such that the value of $|\sum_{ij} a_{ij} x_i x_j|$ is at least a ρ-fraction of the maximum possible value of this quantity (or even the quantity $|\sum_{ij} a_{ij} u_i \cdot u_j|$, where u_i, u_j are unit vectors in a Hilbert space.) Note that here we do not assume that A is positive semidefinite (but we try to maximize the absolute value of the quadratic form, not the quadratic form itself). By applying this approximation algorithm to a matrix defined from a graph G as in the first comment of this section, we can find an induced subgraph that approximates the maximum possible deviation of the total number of edges from its expected value among all induced subgraphs of the graph.

- There is a lot known on the complex version of Grothendieck's Inequality [12, 16], and it may be interesting to find algorithmic or combinatorial applications of these results.

- We believe that the method described in this paper will find further applications, as it provides a rounding technique that can, at least in the cases considered here, handle cancellations between positive and negative terms.
- It will be interesting to improve the approximation guarantee of our algorithms. It will also be interesting to find a ρ-approximation algorithm for the CUT NORM problem (for any absolute positive constant ρ) which does not apply semidefinite programming and/or can be analyzed without relying on Grothendieck's Inequality.

Acknowledgment: Part of this work was carried out during a visit of the first author at Microsoft Research, Redmond, WA, and he thanks his hosts at Microsoft for their hospitality.

References

[1] N. Alon, L. Babai and A. Itai, A fast and simple randomized parallel algorithm for the maximal independent set problem, J. Algorithms 7(1986), 567-583.

[2] N. Alon, W. F. de la Vega, R. Kannan and M. Karpinski, Random sampling and approximation of MAX-CSP problems, Proc. of the 34^{th} ACM STOC, ACM Press (2002), 232–239.

[3] N. Alon, R. A. Duke, H. Lefmann, V. Rödl and R. Yuster, The algorithmic aspects of the Regularity Lemma, Proc. 33^{rd} IEEE FOCS, Pittsburgh, IEEE (1992), 473-481. Also: J. of Algorithms 16 (1994), 80-109.

[4] N. Alon, Y. Matias and M. Szegedy, The space complexity of approximating the frequency moments, Proc. of the 28^{th} ACM STOC, ACM Press (1996), 20-29. Also; J. Comp. Sys. Sci. 58 (1999), 137-147.

[5] N. Alon and J. Spencer, *The Probabilistic Method*, Second Edition, Wiley, New York, 2000.

[6] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, Proof verification and intractability of approximation problems, Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science (FOCS), IEEE Computer Society Press, 1992, 14–23.

[7] A. M. Frieze and R. Kannan, Quick Approximation to matrices and applications, Combinatorica **19** (2) (1999) pp 175–200.

[8] U. Feige and M. Langberg, The RPR^2 Rounding Technique for Semidefinite Programs, Proceedings of the 28th International Colloquium on Automata, Languages and Programming, Crete, Greece, 2001, 213–224.

[9] M. Grötschel, L. Lovász and A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, Combinatorica 1 (1981), 169–197.

[10] A. Grothendieck, Résumé de la théorie métrique des produits tensoriels topologiques, Bol. Soc. Mat. Sao Paulo 8 (1953), 1–79.

[11] M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, J. ACM 42 (1995), 1115–1145.

[12] U. Haagerup, A new upper bound for the complex Grothendieck constant, Israel Journ. Math. 60 (1987), 199–224.

[13] J. Håstad, Some optimal inapproximability results, J. ACM 48 (2001), 798–859.

[14] G. J. O. Jameson, Summing and nuclear norms in Banach space theory, London Mathematical Society Student Texts, 8, Cambridge University Press, Cambridge, 1987.

[15] Johnson, W. B., Lindenstrauss, J., Basic concepts in the geometry of Banach spaces. Handbook of the geometry of Banach spaces, Vol. I, 1–84, North-Holland, Amsterdam, 2001.

[16] H. König, On the complex Grothendieck constant in the n-dimensional case, London Math. Soc. Lect. Notes 158 (1990), 181–198.

[17] J. L. Krivine, Sur la constante de Grothendieck, C. R. Acad. Sci. Paris Ser. A-B 284 (1977), 445–446.

[18] M. Lewin, D. Livnat and U. Zwick, Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems, Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization, Cambridge, Massachusetts, 2002, 67–82.

[19] J. Lindenstrauss and A. Pełczyński, Absolutely summing operators in $\mathrm{L_p}$-spaces and their applications, Studia Math. 29 (1968), 275-326.

[20] S. Mahajan and H. Ramesh, Derandomizing approximation algorithms based on semidefinite programming, SIAM J. Comput. 28 (1999), 1641–1663.

[21] Y. E. Nesterov, Semidefinite relaxation and nonconvex quadratic optimization, Optimization Methods and Software 9 (1998), 141–160.

[22] C. H. Papadimitriou and M. Yannakakis, Optimization, approximation, and complexity classes, J. Comp. Sys. Sci. 43 (1991), 425–440.

[23] R. E. Rietz, A proof of the Grothendieck inequality, Israel J. Math. 19 (1974), 271–276.

[24] E. Szemerédi, *Regular partitions of graphs*, in: *Proc. Colloque Inter. CNRS* (J. -C. Bermond, J. -C. Fournier, M. Las Vergnas and D. Sotteau eds.), 1978, 399-401.

[25] U. Zwick, Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to MAX CUT and other problems, Proceedings of the 31th Annual ACM Symposium on Theory of Computing (STOC), Atlanta, Georgia, 1999, 679–687.