Separating MAX 2-AND, MAX DI-CUT and MAX CUT

Joshua Brakensiek
Computer Science Department
Stanford University
Stanford, CA, USA
jbrakens@cs.stanford.edu

Neng Huang
Department of Computer Science
University of Chicago
Chicago, IL, USA
nenghuang@uchicago.edu

Aaron Potechin
Department of Computer Science
University of Chicago
Chicago, IL, USA
potechin@uchicago.edu

Uri Zwick
Blavatnik School of Computer Science
Tel Aviv University
Tel Aviv, Israel
zwick@tau.ac.il

***Abstract*—Assuming the Unique Games Conjecture (UGC), the best approximation ratio that can be obtained in polynomial time for the MAX CUT problem is $\alpha_{\text{CUT}} \simeq 0.87856$, obtained by the celebrated SDP-based approximation algorithm of Goemans and Williamson. Currently, the best approximation algorithm for MAX DI-CUT, i.e., the MAX CUT problem in *directed* graphs, achieves a ratio of about 0.87401, leaving open the question whether MAX DI-CUT can be approximated as well as MAX CUT. We obtain a slightly improved algorithm for MAX DI-CUT and a new UGC-hardness result for it, showing that $0.87446 \leq \alpha_{\text{DI-CUT}} \leq 0.87461$, where $\alpha_{\text{DI-CUT}}$ is the best approximation ratio that can be obtained in polynomial time for MAX DI-CUT under UGC. The new upper bound separates MAX DI-CUT from MAX CUT, i.e., shows that MAX DI-CUT cannot be approximated as well as MAX CUT, resolving a question raised by Feige and Goemans.**

A natural generalization of MAX DI-CUT is the MAX 2-AND problem in which each constraint is of the form $z_1 \wedge z_2$, where z_1 and z_2 are literals, i.e., variables or their negations. (In MAX DI-CUT each constraint is of the form $\bar{x}_1 \wedge x_2$, where x_1 and x_2 are variables.) Austrin separated MAX 2-AND from MAX CUT by showing that $\alpha_{\text{2AND}} \leq 0.87435$ and conjectured that MAX 2-AND and MAX DI-CUT have the same approximation ratio. Our new lower bound on MAX DI-CUT refutes this conjecture, completing the separation of the three problems MAX 2-AND, MAX DI-CUT and MAX CUT. We also obtain a new lower bound for MAX 2-AND showing that $0.87414 \leq \alpha_{\text{2AND}} \leq 0.87435$.

Our upper bound on MAX DI-CUT is achieved via a simple analytical proof. The new lower bounds on MAX DI-CUT and MAX 2-AND, i.e., the new approximation algorithms, use experimentally-discovered distributions of rounding functions which are then verified via computer-assisted proofs.[1]

***Index Terms*—approximation algorithms, hardness of approximation, constraint satisfaction problem, maximum cut, semidefinite programming, computer-assisted proof**

The first author is support in part by an NSF Graduate Research Fellowship and a Microsoft Research PhD Fellowship. The second and third authors are supported in part by NSF grant CCF:2008920. We thank anonymous referees for various helpful comments.

[1]Code for the project: https://github.com/jbrakensiek/max-dicut

I. Introduction

Goemans and Williamson [1], in their seminal paper, introduced the paradigm of obtaining approximation algorithms for Boolean *Constraint Satisfaction Problems* (CSPs) by first obtaining a semidefinite programming (SDP) *relaxation* of the problem and then *rounding* an optimal solution of the relaxation. The first, and perhaps biggest, success of this paradigm is a simple and elegant α_{GW}-approximation algorithm, where $\alpha_{\text{GW}} \simeq 0.87856$, for the MAX CUT problem, i.e., the maximum cut problem in undirected graphs, improving for the first time over the naive $\frac{1}{2}$-approximation algorithm. Goemans and Williamson [1] also obtained improved algorithms for the MAX DI-CUT, MAX 2-SAT and MAX SAT problems.

Feige and Goemans [2], Matuura and Matsui [3] and Lewin, Livnat and Zwick [4] obtained improved approximation algorithms for the MAX 2-SAT and MAX DI-CUT problems. The best approximation ratios, obtained by Lewin, Livnat and Zwick [4], are 0.940 for MAX 2-SAT and 0.874 for MAX DI-CUT. Karloff and Zwick [5] obtained an optimal (see below) $\frac{7}{8}$-approximation algorithm for MAX $\{1,2,3\}$-SAT and Zwick [6] obtained approximation algorithms, some of them optimal, for many other MAX 3-CSP problems, i.e., maximization versions of Boolean CSP problems in which each constraint is on at most three variables. Andersson and Engebretsen [7], Zwick [8], Halperin and Zwick [9], Asano and Williamson [10], Zhang, Ye and Han [11], and Avidor, Berkovitch and Zwick [12] obtained approximation algorithms for various versions of the MAX SAT and MAX NAE-SAT problems. It is a major open problem whether there is a $\frac{7}{8}$-approximation algorithm for the MAX SAT problem. Brakensiek, Huang, Potechin and Zwick [13] showed that there is no $\frac{7}{8}$-approximation algorithm for the MAX NAE-SAT problem, assuming UGC. Abbasi-Zadeh et al [14] and Eldan and Naor [15] used "sticky Brownian motion" to obtain optimal, or close to optimal, algorithms for MAX CUT and related problems. For a survey of these and related results, see Makarychev and Makarychev [16].

Håstad [17], in a major breakthrough, extending the celebrated PCP theorem of Arora et al. [18], showed, among other things, that, for any $\varepsilon > 0$, it is NP-hard to obtain a $(\frac{7}{8} + \varepsilon)$-approximation of MAX 3-SAT and a $(\frac{1}{2} + \varepsilon)$-approximation of MAX 3-LIN, showing that the trivial algorithms for these two problems that just choose a random assignment are tight. Trevisan, Sorkin, Sudan and Williamson [19] showed, using gadget reductions, that it is NP-hard to obtain a $(\frac{16}{17} + \varepsilon)$-approximation of MAX CUT and $(\frac{12}{13} + \varepsilon)$-approximation of MAX DI-CUT.

Khot [20] introduced the *Unique Games Conjecture* (UGC). Khot, Kindler, Mossel and O'Donnell [21] then showed that UGC and the Majority is Stablest Conjecture (later proved by Mossel, O'Donnell and Oleszkiewicz [22]) implies that, for any $\varepsilon > 0$, obtaining an $(\alpha_{\text{GW}} + \varepsilon)$-approximation for MAX CUT is NP-hard, showing, quite remarkably, that the algorithm of Goemans and Williamson [1] is optimal, i.e., $\alpha_{\text{CUT}} = \alpha_{\text{GW}}$, assuming UGC. Austrin [23] then showed that the MAX 2-SAT algorithm of Lewin, Livnat and Zwick [4] is essentially optimal, again modulo UGC. Austrin [24] obtained some upper bounds on the approximation ratio that can be achieved for MAX 2-AND in polynomial time. However, they do not match the approximation ratio obtained by the MAX DI-CUT algorithm of Lewin, Livnat and Zwick [4] which is in fact an approximation algorithm for MAX 2-AND.

Raghavendra [25], [26], in another breakthrough, showed that under UGC, the best approximation ratio that can be obtained for any MAX CSP problem, over a finite domain and with a finite number of constraint types, can be obtained using a canonical SDP relaxation of the problem and the rounding of an optimal solution of this relaxation using an appropriate rounding procedure taken from a specified family of rounding procedures. The approximation ratio obtained is then exactly the *integrality gap* of the relaxation. Approximating the integrality gap up to ε takes doubly exponential time in $1/\varepsilon$, and a close to optimal algorithm can be obtained by trying discretized versions of all rounding procedures, up to some resolution. (See Raghavendra and Steurer [27] for more on finding almost optimal rounding schemes.)

It might seem that these results resolve all problems related to the approximation of MAX CSP problems. Unfortunately, this is not the case. These results do give valuable guidance to the designers of approximation algorithms. In particular, it is clear which semidefinite programming relaxation should be used and the search for an optimal, or almost optimal, rounding procedure can be restricted to the family of rounding procedures specified by Raghavendra [25]. However, these results give almost no concrete information on the integrality gap of the relaxation, which is also the best approximation ratio that can be obtained. Also, no practical information is given on how to obtain optimal, or almost optimal rounding procedures, other than the fact that they belong to a huge class of rounding procedures, as it is wildly impractical to implement and run a brute force algorithm whose running time is doubly exponential in $1/\varepsilon$.

In particular, Raghavendra's results are unable[2] to answer the following questions: Is there a $\frac{7}{8}$-approximation algorithm for MAX SAT, with clauses of all sizes allowed? Can MAX DI-CUT be approximated as well as MAX CUT? Can MAX 2-AND be approximated as well as MAX DI-CUT? In this paper we study the latter two questions and answer them in the negative, assuming UGC.

A. *Our results*

Our main result is the following theorem.

Theorem I.1 (Main)**.** *Assuming UGC,* $\alpha_{\mathit{2AND}} < \alpha_{\mathit{DI\text{-}CUT}} < \alpha_{\mathit{CUT}}$.

To separate MAX 2-AND, MAX DI-CUT and MAX CUT, we obtain an improved upper bound and an improved lower bound (i.e., an approximation algorithms) for MAX DI-CUT. Our improved upper bound is:

Theorem I.2. *Assuming UGC,* $\alpha_{\mathit{DI\text{-}CUT}} \leq 0.87461$.

To obtain the new upper bound, we construct a distribution over MAX DI-CUT *configurations* that is hard for any rounding procedure from the family $\mathcal{THRESH}^-$ defined by Lewin, Livnat and Zwick [4]. Such hard distributions can then be converted into *dictatorship tests* and then Unique Games hardness by small modifications to the technique used by Austrin [24] for distributions over MAX 2-AND configurations.

The reason why MAX 2-AND and MAX DI-CUT have a lower approximation ratio than MAX CUT is as follows. By symmetry, for instances of MAX CUT, there is always an SDP solution where none of the variables have any bias towards -1 or 1. Thus, for MAX CUT, the rounding scheme only needs to use the pairwise biases between variables and it turns out that hyperplane rounding is the optimal way to do this. However, for MAX 2-AND and MAX DI-CUT, the SDP solution may have variables with nonzero biases. In this case, there are some configurations where it is better to focus more on the biases while for other configurations it is better to focus more on the pairwise biases. Mixing these configurations gives a harder distribution of configurations.

That said, it is more difficult to obtain hard configurations for MAX DI-CUT than for MAX 2-AND, since in MAX 2-AND the functions used in the rounding scheme can be assumed, without loss of generality, to be *odd*. (A function $f : [-1, 1] \to \mathbb{R}$ is odd if and only if $f(-x) = -f(x)$ for every $x \in [-1, 1]$.) Using an odd rounding function ensures that a variable and its negation are assigned opposite truth values. In MAX DI-CUT there is no such restriction as, in a sense, there are no negated variables. The possibility of using non-odd rounding functions gives the rounding scheme more power. (The improved rounding scheme that we obtain for MAX DI-CUT uses a distribution of

[2]In particular, if the answer to any of these questions is "yes" the Raghavendra-Steurer algorithm cannot certify these in finite time, and if the answer is "no" the ε needed for separation is so small that the algorithm would need to run over a galactic time scale.

rounding functions some of which are not odd. This is exactly what enables the separation of MAX DI-CUT from MAX 2-AND, as we discuss below.) We overcome this difficulty using a simple, symmetric construction for which the best rounding scheme is odd. Another interesting feature of our hard construction is that it contains a configuration for which all the triangle inequalities, powerful constraints of the SDP relaxation, are not tight. This is in contrast to previous work on MAX 2-SAT [23] and MAX 2-AND [24], where hardness results are derived only from configurations in which one of the triangle inequalities is tight.

Our construction yields an upper bound of $\alpha_{\text{DI-CUT}} \leq 0.87461$, which together with $\alpha_{\text{CUT}} \geq 0.87856$ exhibits a clear separation between MAX DI-CUT and MAX CUT. (Although the separation is clear, it is still perplexing that the approximation ratios of MAX CUT and MAX DI-CUT are so close, and yet not equal.) We believe that our upper bound can be slightly improved using a sequence of more and more complicated constructions that yield slightly better and better upper bounds.

In addition to our improved upper bound for MAX DI-CUT, we also obtain two new lower bounds for MAX DI-CUT and MAX 2-AND.

Theorem I.3. $\alpha_{\textit{DI-CUT}} \geq 0.87446$. *(In other words, there is an approximation algorithm for* MAX DI-CUT *with an approximation ratio of at least* 0.87446.*)*

Theorem I.4. $\alpha_{\textit{2AND}} \geq 0.87414$. *(In other words, there is an approximation algorithm for* MAX 2-AND *with an approximation ratio of at least* 0.87414.*)*

The new lower bounds improve on the previously best, and non-rigorous, bound of 0.87401 obtained by Lewin, Livnat and Zwick [4] for both MAX DI-CUT and MAX 2-AND. Despite the relatively small improvements, the improved approximation algorithms are interesting for at least two reasons. The first is that the new approximation algorithm for MAX DI-CUT separates MAX DI-CUT from MAX 2-AND, refuting a conjecture of Austrin [24]. The second is that the new algorithms show that taking a single rounding scheme from $\mathcal{THRESH}^-$, as done by Lewin, Livnat and Zwick [4] and as shown by Austrin [23], [24] to be sufficient for obtaining an optimal approximation algorithm for MAX 2-SAT, is not sufficient for obtaining optimal approximation algorithms for MAX DI-CUT and MAX 2-AND. Using insights gained from the upper bounds, we design an improved approximation algorithm for MAX DI-CUT that uses *distributions* of $\mathcal{THRESH}^-$ rounding procedures, i.e., rounding procedures belonging to the more general family $\mathcal{THRESH}$ also defined in Lewin, Livnat and Zwick [4]. Using a computer search, we find a new rounding procedure from this family[3] which shows that $\alpha_{\text{DI-CUT}} \geq 0.87446$. Our rigorous proof of this inequality is computer assisted.

In [4], the authors discovered their $\mathcal{THRESH}^-$ procedures for MAX DI-CUT and MAX 2-AND using non-convex optimization. More precisely, they used a local descent procedure from random starting points to tune a single rounding function that performs well for all possible configurations simultaneously. However, this approach becomes impractical for finding an optimal probability distribution of $\mathcal{THRESH}^-$ functions (i.e., a "$\mathcal{THRESH}$ scheme"). One potential reason why this would not work is that there would be a significant local optimum where all the functions in the distribution identical to the one in Lewin, Livnat and Zwick [4].

Instead, we cast the design of the $\mathcal{THRESH}$ scheme for MAX DI-CUT and MAX 2-AND as infinite *zero-sum games* played by two players. The first player, Alice, selects a $\mathcal{THRESH}^-$ function and the second player, Bob, selects a configuration of SDP vectors to round. (This configuration may or may not correspond to an optimal solution of an SDP relaxation of an actual instance.) Alice's value is then the approximation ratio achieved by her $\mathcal{THRESH}^-$ function on the SDP value of the configuration. Bob's value is the negative of Alice's value. One can then show, that $\alpha_{\text{DI-CUT}}$ (or α_{2AND}) is precisely the value of this game, assuming UGC and the positivity conjecture in the work of Austrin [24]. Computationally, we discretize this game and use a min-max optimization procedure to estimate the value of this game and find an optimal, or almost optimal, strategy for Alice. This proceeds in a series of phases: Bob challenges Alice with a distribution of instances, and Alice computes a nearly-optimal response using methods similar to that of Lewin, Livnat and Zwick [4]. Then, with Alice's functions, Bob computes a new distribution of instances which Alice does the worst one. This latter step is done by solving a suitable LP (the dual variables tell us Alice's optimal $\mathcal{THRESH}$ scheme). As the "raw" $\mathcal{THRESH}$ scheme produced by this procedure can be somewhat noisy, we subsequently manually simplified the $\mathcal{THRESH}$ distribution.

As mentioned, the proofs of the bounds $\alpha_{\text{DI-CUT}} \geq 0.87446$ and $\alpha_{\text{2AND}} \geq 0.87414$ are computer-assisted, using the technique of *interval arithmetic*. This technique has been previously used in the study of approximation algorithms. For example, Zwick [28] used it to certify the $\frac{7}{8}$-approximation ratio for MAX {1,2,3}-SAT claimed by Karloff and Zwick [5]. The use of interval arithmetic in our setting is much more challenging, however, as the rounding procedures used for MAX DI-CUT are much more complicated than the simple random hyperplane rounding used for MAX {1,2,3}-SAT. In particular, we need to use rigorous numerical integration to compute two-dimensional normal probabilities. A computer-assisted verification is probably necessary in our setting since fairly complicated distributions seem to be needed for obtaining good approximation ratios, and it is hard to imagine that such distributions can be analyzed manually.

Since Austrin [24] showed that $\alpha_{\text{2AND}} < 0.87435$, assuming UGC, our new MAX DI-CUT approximation algorithm separates MAX 2-AND and MAX DI-CUT. This refutes Austrin's conjecture that MAX 2-AND and MAX DI-CUT have the same approximation ratios. It also gives an interesting, non-trivial, example where a positive CSP (i.e., CSP that does not allow negated variables)

[3]Technically, we add in a tiny amount of independent rounding for verification purposes but we believe this can be removed.

is strictly easier to approximate than the CSP with the same predicate when negated variables are allowed.

We believe that the fact that rounding procedures from $\mathcal{THRESH}^-$ do not yield optimal approximation algorithms for MAX DI-CUT is interesting in its own right. We conjecture that distributions over such procedures, i.e., rounding procedures from $\mathcal{THRESH}$ are enough to obtain optimal algorithms for MAX DI-CUT and MAX 2-AND. (A continuous distribution is probably needed to get the optimal algorithms.)

We note that both $\mathcal{THRESH}^-$ and $\mathcal{THRESH}$ are tiny subfamilies of the families shown by Raghavendra [25] to be enough for obtaining optimal approximation algorithms for general MAX CSP problems. In particular, $\mathcal{THRESH}^-$ and $\mathcal{THRESH}$ use only one Gaussian random vector while, in general, the families of Raghavendra [25] may need an unbounded number of such random vectors to obtain optimal or close-to-optimal results.

B. Organization of the paper

The rest of the paper is organized as follows. In Section II we introduce the MAX CUT, MAX DI-CUT and MAX 2-AND problems and their SDP relaxations, we state the Unique Games Conjecture, and we introduce the $\mathcal{THRESH}^-$ and $\mathcal{THRESH}$ families of rounding procedures used throughout the paper. In Section III we derive our new upper bound on MAX DI-CUT which separates MAX DI-CUT from MAX CUT. The proof of this upper bound is completely analytical. In Section IV we describe the computation techniques used to discover our improved MAX DI-CUT algorithm and the computation techniques used to rigorously verify the approximation ratio that it achieves. In Section V we obtain corresponding results for the MAX 2-AND problem. We end in Section VI with some concluding remarks and open problems.

II. Preliminaries

A. MAX CSP and canonical SDP relaxations

For a Boolean variable, we associate -1 with true and 1 with false. A Boolean predicate on k variables is a function $P : \{-1,1\}^k \to \{0,1\}$. If P outputs 1, then we say P is satisfied.

Definition II.1 (MAX CSP(P)). *Let P be a Boolean predicate on k variables. An instance of* MAX CSP(P) *is defined by a set of Boolean variables $\mathcal{V} = \{x_1, x_2, \ldots, x_n\}$ and a set of constraints $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$, where each constraint C_i is of the form $P(b_{i,1}x_{j_{i,1}}, b_{i,2}x_{j_{i,2}}, \ldots, b_{i,k}x_{j_{i,k}})$ for some $j_{i,1}, \ldots, j_{i,k} \in [n]$ and $b_{i,1}, b_{i,2}, \ldots b_{i,k} \in \{-1,1\}$, and a weight function $w : \mathcal{C} \to [0,1]$ satisfying $\sum_{i=1}^m w(C_i) = 1$. The goal is to find an assignment to the variables that maximizes $\sum_{i=1}^m w(i)P(b_{i,1}x_{j_{i,1}}, b_{i,2}x_{j_{i,2}}, \ldots, b_{i,k}x_{j_{i,k}})$, i.e., the sum of the weights of satisfied constraints.*

Definition II.2 (MAX CSP$^+(P)$). MAX CSP$^+(P)$ *has the same definition as* MAX CSP(P)*, except that now each constraint C_i is of the form $P(x_{j_{i,1}}, x_{j_{i,2}}, \ldots, x_{j_{i,k}})$. In other words, negated variables are not allowed.*

Since the weight function is non-negative and sums up to 1, we can think of it as a probability distribution over the constraints. Note that we only defined CSPs with a single Boolean predicate, while in general there can be more than one predicate and they may not be Boolean. We refer to a CSP with a k-ary predicate as a k-CSP.

We are now ready to define the three MAX 2-CSP problems that we separate.

Definition II.3. *Let* CUT $: \{-1,1\}^2 \to \{0,1\}$ *be the predicate which is satisfied if and only if the two inputs are not equal. Let* DI-CUT $: \{-1,1\}^2 \to \{0,1\}$ *be the predicate which is satisfied if and only if $x = 1$ and $y = -1$. Then* MAX CUT *is the problem* MAX CSP$^+$(CUT)*,* MAX DI-CUT *is the problem* MAX CSP$^+$(DI-CUT) *and* MAX 2-AND *is the problem* MAX CSP(DI-CUT).

In graph-theoretic language, we can think of each variable in a MAX DI-CUT instance as a vertex, and each constraint as a weighted direct edge between two vertices. An assignment of $+1$ and -1 to the vertices defines a directed cut in the graph. We are asked to assign $+1$ and -1 to the vertices so that the sum of the weights of edges that cross the cut, i.e., go from $+1$ to -1, is maximized.

We can also define AND $: \{-1,1\}^2 \to \{0,1\}$ such that AND$(x,y) = 1$ if and only if $x = y = -1$. Note that then DI-CUT$(x,y) =$ AND$(\bar{x},y)$, and MAX 2-AND is also MAX CSP(AND), hence its name.

The following Fourier expansion of DI-CUT is heavily used throughout the paper.

Proposition II.4. DI-CUT$(x,y) = \frac{1+x-y-xy}{4}$.

This proposition can be used to extend the domain of DI-CUT to real inputs.

Any MAX CSP(P) has a *canonical* semi-definite programming relaxation. The canonical SDP relaxation for MAX DI-CUT, for example, is:

$$\text{Maximize} \quad \sum_{C=\text{DI-CUT}(x_i,x_j)\in\mathcal{C}} w_C \cdot \frac{1+\mathbf{v}_0\cdot\mathbf{v}_i-\mathbf{v}_0\cdot\mathbf{v}_j-\mathbf{v}_i\cdot\mathbf{v}_j}{4}$$

$$\text{subject to}$$

$$\forall i \in \{0,1,2,\ldots,n\}, \qquad \mathbf{v}_i\cdot\mathbf{v}_i = 1,$$

$$\forall C = \text{DI-CUT}(x_i,x_j) \in \mathcal{C}, \quad \begin{array}{rcl} (\mathbf{v}_0-\mathbf{v}_i)\cdot(\mathbf{v}_0-\mathbf{v}_j) & \ge & 0, \\ (\mathbf{v}_0+\mathbf{v}_i)\cdot(\mathbf{v}_0-\mathbf{v}_j) & \ge & 0, \\ (\mathbf{v}_0-\mathbf{v}_i)\cdot(\mathbf{v}_0+\mathbf{v}_j) & \ge & 0, \\ (\mathbf{v}_0+\mathbf{v}_i)\cdot(\mathbf{v}_0+\mathbf{v}_j) & \ge & 0. \end{array}$$

The canonical SDP relaxation is obtained as follows. There is a unit vector $\mathbf{v}_i \in \mathbb{R}^{n+1}$ for each variable x_i, and a special unit vector $\mathbf{v}_0$ corresponding to false. Each linear term x_i in the Fourier expansion of P is replaced by $\mathbf{v}_0 \cdot \mathbf{v}_i$, and each quadratic term $x_i x_j$ is replaced by $\mathbf{v}_i \cdot \mathbf{v}_j$. The so-called triangle inequalities are then added.

Note that this is the special case of Raghavendra's basic SDP in the setting of Boolean 2-CSPs, and the triangle inequalties ensure that there is a local distribution of assignments for each constraint.

B. Unique Games Conjecture

The Unique Games Conjecture (UGC), introduced by Khot [20], plays a crucial role in the study of hardness of approximation of CSPs. One version of the conjecture is as follows.

Definition II.5 (Unique Games)**.** *In a unique games instance $I = (G, L, \Pi)$, we are given a weighted graph $G = (V(G), E(G), w)$, a set of labels $[L] = \{1, 2, \ldots, L\}$ and a set of permutations $\Pi = \{\pi_e^v : [L] \to [L] \mid e = \{v, u\} \in E(G)\}$ such that for every $e = \{u, v\} \in E(G)$, $\pi_e^v = (\pi_e^u)^{-1}$. An assignment to this instance is a function $A : V(G) \to [L]$. We say that A satisfies an edge $e = \{u, v\}$ if $\pi_e^u(A(u)) = A(v)$. The value of an assignment A is the weight of satisfied edges, i.e., $\mathrm{Val}(I, A) = \sum_{e\in E(G): A \text{ satisfies } e} w(e)$, and the value of the instance $\mathrm{Val}(I)$ is defined to be the value of the best assignment, i.e., $\mathrm{Val}(I) = \max_A \mathrm{Val}(I, A)$.*

Conjecture (Unique Games Conjecture)**.** *For any $\eta, \gamma > 0$, there exists a sufficiently large L such that the problem of determining whether a given unique games instance I with L labels has $\mathrm{Val}(I) \geq 1 - \eta$ or $\mathrm{Val}(I) \leq \gamma$ is NP-hard.*

We say that a problem is UG-hard, if it is NP-hard assuming the UGC. Raghavendra [25] showed that any integrality gap instance of the canonical SDP relaxation can be turned into a UG-hardness result.

C. Configurations of biases and pairwise biases

As it turns out, an actual integrality gap instance is not required to derive UG-hardness results. Instead, it is sufficient to consider configurations of SDP solution vectors that appear in the same constraint. For 2-CSPs, each such configuration is represented by a triplet $\theta = (b_i, b_j, b_{ij})$, where $b_i = \mathbf{v}_0 \cdot \mathbf{v}_i$, and $b_j = \mathbf{v}_0 \cdot \mathbf{v}_j, b_{ij} = \mathbf{v}_i \cdot \mathbf{v}_j$. b_i and b_j are called *biases* and b_{ij} is called a *pairwise bias*. A valid configuration is required to satisfy the triangle inequalities described in the previous section. We will use Θ for a set of valid configurations, and $\tilde{\Theta}$ for such a set endowed with a probability distribution.

Definition II.6 (Completeness)**.** *Given a configuration $\theta = (b_i, b_j, b_{ij})$ for MAX DI-CUT, its completeness is defined as $\mathsf{Comp}(\theta) = \frac{1+b_i-b_j-b_{ij}}{4}$. For a distribution of configurations $\tilde{\Theta}$, its completeness is defined as $\mathsf{Comp}(\tilde{\Theta}) = \mathbb{E}_{\theta\sim\tilde{\Theta}}[\mathsf{Comp}(\theta)]$.*

Note that if $\tilde{\Theta}$ actually comes from an SDP solution, then $\mathsf{Comp}(\tilde{\Theta})$ is simply the SDP value of this solution.

Definition II.7 (Relative pairwise bias)**.** *Given a configuration $\theta = (b_i, b_j, b_{ij})$, the relative pairwise bias is defined as $\rho(\theta) = \frac{b_{ij}-b_i b_j}{\sqrt{(1-b_i^2)(1-b_j^2)}}$, if $(1 - b_i^2)(1 - b_j^2) \neq 0$, and 0 otherwise.*

Geometrically, $\rho(\theta)$ is the inner product between $\mathbf{v}_i$ and $\mathbf{v}_j$ after removing their components parallel to $\mathbf{v}_0$ and renormalizing.

Definition II.8 (Positive configurations [24])**.** *Given a Boolean predicate $P(x_1, x_2)$ on two variables with Fourier expansion $\frac{\hat{P}_\emptyset+\hat{P}_1 x_1+\hat{P}_2 x_2+\hat{P}_{1,2} x_1 x_2}{4}$, a configuration $\theta = (b_i, b_j, b_{ij})$ for MAX CSP(P) (or MAX CSP$^+(P)$) is called* positive *if $\hat{P}_{1,2} \cdot \rho(\theta) \geq 0$.*

If $P =$ DI-CUT, then the quadratic coefficient in the Fourier expansion is $-1/4$, which implies that a configuration is positive if and only if its relative pairwise bias is not positive. Austrin [24] presented a general mechanism to deduce UG-hardness results for MAX CSP(P) from hard distributions of positive configurations. With very slight modifications, the same mechanism can also be used for MAX CSP$^+(P)$. Austrin also conjectured that positive configurations are the hardest to round. This conjecture is still open. Our results do not rely on this conjecture.

In a MAX DI-CUT instance, if we flip the direction of every edge in the graph, then an optimal solution to this new instance can be obtained by flipping all the signs in an optimal solution to the original instance. For configurations, this symmetry corresponds to swapping the two biases and then changing the signs.

Definition II.9 (Flipping a configuration)**.** *Let $\theta = (b_i, b_j, b_{ij})$ be a DI-CUT configuration. We define its* flip *to be $\mathrm{flip}(\theta) = (-b_j, -b_i, b_{ij})$.*

The following proposition can be easily verified.

Proposition II.10. *Let $\theta = (b_i, b_j, b_{ij})$ be a DI-CUT configuration. We have*

1) $\rho(\theta) = \rho(\mathrm{flip}(\theta))$.
2) $\mathsf{Comp}(\theta) = \mathsf{Comp}(\mathrm{flip}(\theta))$.

D. The $\mathcal{THRESH}$ and $\mathcal{THRESH}^-$ families of rounding functions

$\mathcal{THRESH}$ and $\mathcal{THRESH}^-$, first introduced in Lewin, Livnat and Zwick [4], are small but powerful families of rounding functions for SDP relaxations of CSPs. In a $\mathcal{THRESH}^-$ rounding scheme, a continuous threshold function $f : [-1, 1] \to \mathbb{R}$ is specified. The algorithm chooses a random Gaussian vector $\mathbf{r} \in \mathbb{R}^{n+1}$, and sets each variable x_i to true (-1) if and only if $\mathbf{r} \cdot \mathbf{v}_i^\perp \geq f(\mathbf{v}_0 \cdot \mathbf{v}_i)$, where

$$\mathbf{v}_i^\perp = \frac{\mathbf{v}_i - (\mathbf{v}_i \cdot \mathbf{v}_0)\mathbf{v}_0}{\sqrt{1 - (\mathbf{v}_i \cdot \mathbf{v}_0)^2}}$$

is the component of $\mathbf{v}_i$ orthogonal to $\mathbf{v}_0$ renormalized to a unit vector. (If $\mathbf{v}_i = \pm\mathbf{v}_0$, we can take $\mathbf{v}_i^\perp$ to be any unit vector that is orthogonal to every other vector in the SDP solution.) Since $\mathbf{v}_i^\perp$ is a unit vector, $\mathbf{r}\cdot\mathbf{v}_i^\perp$ is a standard normal random variable. Furthermore, for any $i,j\in[n]$, $\mathbf{r}\cdot\mathbf{v}_i^\perp$ and $\mathbf{r}\cdot\mathbf{v}_j^\perp$ are jointly Gaussian with correlation $\mathbf{v}_i^\perp\cdot\mathbf{v}_j^\perp$.

Let Φ,φ be the c.d.f. and p.d.f. of the standard normal distribution, respectively. For $t_1,t_2\in\mathbb{R}$, let $\Phi_\rho(t_1,t_2):=\Pr[X\le t_1\wedge Y\le t_2]$, where X and Y are two standard normal random variables that are jointly Gaussian with $\mathbb{E}[XY]=\rho$. Then for a $\mathcal{THRESH}^-$ rounding scheme with threshold function f, a variable x_i is rounded to false with probability $\Phi(f(b_i))$. For a DI-CUT configuration $\theta=(b_i,b_j,b_{ij})$, the probability that it is satisfied by $\mathcal{THRESH}^-$ with f, which happens when x_i is set to false and x_j is set to true, is equal to

$$\begin{aligned}&\Pr\left[\mathbf{r}\cdot\mathbf{v}_i^\perp\le f(b_i)\text{ and }\mathbf{r}\cdot\mathbf{v}_j^\perp\ge f(b_j)\right]\\ =\;&\Pr\left[\mathbf{r}\cdot\mathbf{v}_i^\perp\le f(b_i)\text{ and }-\mathbf{r}\cdot\mathbf{v}_j^\perp\le -f(b_j)\right]\\ =\;&\Phi_{-\rho(\theta)}(f(b_i),-f(b_j))\ .\end{aligned}$$

This naturally leads to the following definition.

Definition II.11 (Soundness)**.** *Let $f:[-1,1]\to\mathbb{R}$ be a continuous threshold function and $\theta=(b_i,b_j,b_{ij})$ a configuration for* MAX DI-CUT*. We define* $\mathsf{Sound}(\theta,f)=\Phi_{-\rho(\theta)}(f(b_i),-f(b_j))$*. For a distribution of configurations $\tilde{\Theta}$, its soundness* $\mathsf{Sound}(\tilde{\Theta},f)$ *is defined as* $\mathbb{E}_{\theta\sim\tilde{\Theta}}[\mathsf{Sound}(\theta,f)]$.

As in the case for configurations, we can also flip a $\mathcal{THRESH}^-$ threshold function.

Definition II.12. *Let $f:[-1,1]\to\mathbb{R}$ be a continuous threshold function. We define $\mathrm{flip}(f)$ as the function $x\mapsto -f(-x)$.*

Proposition II.13. *Let $f:[-1,1]\to\mathbb{R}$ be a continuous threshold function and $\theta=(b_i,b_j,b_{ij})$ a configuration. Then*

$$\mathsf{Sound}(\theta,f)\;=\;\mathsf{Sound}(\mathrm{flip}(\theta),\mathrm{flip}(f))\ .$$

Proof. By Proposition II.10, we have that $\rho(\theta)=\rho(\mathrm{flip}(\theta))=\rho$. By definition of soundness,

$$\begin{aligned}\mathsf{Sound}(\theta,f)&=\Phi_{-\rho}(f(b_i),-f(b_j))\\&=\Phi_{-\rho}(-\mathrm{flip}(f)(-b_i),\mathrm{flip}(f)(-b_j))\\&=\Phi_{-\rho}(\mathrm{flip}(f)(-b_j),-\mathrm{flip}(f)(-b_i))\\&=\mathsf{Sound}(\mathrm{flip}(\theta),\mathrm{flip}(f))\ .\end{aligned}$$ □

A rounding scheme from $\mathcal{THRESH}$ can be thought of as a distribution over $\mathcal{THRESH}^-$ rounding schemes. Formally speaking, a $\mathcal{THRESH}$ rounding scheme is specified by a continuous function $T:\mathbb{R}\times[-1,1]\to\mathbb{R}$, and a variable x_i is set to true if and only if $\mathbf{r}\cdot\mathbf{v}_i^\perp\ge T(\mathbf{v}_0\cdot\mathbf{r},\mathbf{v}_0\cdot\mathbf{v}_i)$. This allows for a continuous distribution over $\mathcal{THRESH}^-$ rounding schemes.

The following partial derivatives are helpful for analyzing $\mathcal{THRESH}$ and $\mathcal{THRESH}^-$ rounding schemes.

Proposition II.14 (Partial derivatives of $\Phi_\rho(t_1,t_2)$)**.**

$$\begin{aligned}\frac{\partial\Phi_\rho(t_1,t_2)}{\partial\rho}&=\frac{1}{2\pi\sqrt{1-\rho^2}}\exp\left(-\frac{t_1^2-2\rho t_1t_2+t_2^2}{2(1-\rho^2)}\right)\ ,\\ \frac{\partial\Phi_\rho(t_1,t_2)}{\partial t_1}&=\varphi(t_1)\Phi\left(\frac{t_2-\rho t_1}{\sqrt{1-\rho^2}}\right)\ ,\\ \frac{\partial\Phi_\rho(t_1,t_2)}{\partial t_2}&=\varphi(t_2)\Phi\left(\frac{t_1-\rho t_2}{\sqrt{1-\rho^2}}\right)\ .\end{aligned}$$

A derivation of the formula given for $\frac{\partial\Phi_\rho(t_1,t_2)}{\partial\rho}$ can be found in Drezner and Wesolowsky [29]. The formulas for $\frac{\partial\Phi_\rho(t_1,t_2)}{\partial t_1}$ and $\frac{\partial\Phi_\rho(t_1,t_2)}{\partial t_2}$ follow easily from the definition of $\Phi_\rho(t_1,t_2)$.

III. Upper bounds for MAX DI-CUT

A. *Separating* MAX DI-CUT *from* MAX CUT

In this section, we prove the following theorem, which separates MAX DI-CUT from MAX CUT.

Theorem III.1. *Assuming the Unique Games Conjecture, it is NP-hard to approximate* MAX DI-CUT *within a factor of* 0.87461.

To prove Theorem III.1 we construct a distribution of positive configurations $\tilde{\Theta}$, compute its completeness, and show that no $\mathcal{THRESH}^-$ rounding scheme can achieve a performance ratio of 0.87461 on it. The UG-hardness result then follows from a slight generalization of a reduction of Austrin [24]. [4] (We describe this reduction in the full version.)

The distribution $\tilde{\Theta}$ used to obtain the upper bound is extremely simple. Let p_1,p_2,b,c be some parameters to be chosen later. We will choose them so that $b,p_1,p_2\in(0,1)$, $c\in(-1,-b^2)$, and $2p_1+p_2=1$. Consider the following distribution of configurations $\tilde{\Theta}=\{\theta_1,\theta_2,\theta_3\}$:

$$\begin{aligned}\theta_1&=(-b,-b,-1+2b)&&\text{with probability }p_1\\ \theta_2&=(\;\;b,-b,\quad c\quad)&&\text{with probability }p_2\\ \theta_3&=(\;\;b,\;\;\;b,-1+2b)&&\text{with probability }p_1\end{aligned}$$

Note that in the θ_1 and θ_3 one of the triangle inequalities is tight, while in θ_2 none of the triangle inequalities are tight, as was mentioned earlier. Also, this distribution is symmetric with respect to flip, since $\mathrm{flip}(\theta_1)=\theta_3$ and $\mathrm{flip}(\theta_2)=\theta_2$.

We first verify that $\tilde{\Theta}$ satisfies the positivity condition.

Proposition III.2. *$\tilde{\Theta}$ is a distribution of positive configurations.*

Proof. In θ_1 and θ_3, the relative pairwise bias is equal to $\rho_1=\frac{-1+2b-b^2}{1-b^2}=-\frac{1-b}{1+b}<0$. In θ_2, the relative pairwise bias is equal to $\rho_2=\frac{c+b^2}{1-b^2}<0$ since we choose $c<-b^2$. □

[4] Since the distribution is fixed, the optimal $\mathcal{THRESH}^-$ rounding scheme for it is also the best $\mathcal{THRESH}$ rounding scheme.

The completeness of this instance can be easily computed.

Proposition III.3.

$$\mathsf{Comp}(\tilde{\Theta}) = p_1 \cdot (1-b) + p_2 \cdot \frac{1+2b-c}{4}.$$

Proof. We have

$$\begin{aligned}
&\mathsf{Comp}(\tilde{\Theta}) \\
&= p_1 \cdot \frac{1+(-b)-(-b)-(-1+2b)}{4} \\
&\quad + p_2 \cdot \frac{1+b-(-b)-c}{4} + p_1 \cdot \frac{1+b-b-(-1+2b)}{4} \\
&= p_1 \cdot \frac{2-2b}{4} + p_2 \cdot \frac{1+2b-c}{4} + p_1 \cdot \frac{2-2b}{4} \\
&= p_1 \cdot (1-b) + p_2 \cdot \frac{1+2b-c}{4}.
\end{aligned}$$

□

We now give an upper bound on the performance of any $\mathcal{THRESH}^-$ rounding scheme on this distribution. Let $t_1 = f(-b)$ and $t_2 = f(b)$ be the thresholds for $-b, b$ respectively. Let $s(t_1, t_2)$ be the soundness of this rounding scheme on $\tilde{\Theta}$. By definition of $\mathcal{THRESH}^-$, we have

$$\begin{aligned}
s(t_1,t_2) \;&=\; p_1 \cdot \Phi_{-\rho_1}(t_1, -t_1) + p_2 \cdot \Phi_{-\rho_2}(t_2, -t_1) \\
&\quad + p_1 \cdot \Phi_{-\rho_1}(t_2, -t_2) \ ,
\end{aligned}$$

where $\rho_1, \rho_2 < 0$ are computed in Proposition III.2. We first look at the case where $-\infty < t_1, t_2 < \infty$. The case where $t_1 = \pm\infty$ or $t_2 = \pm\infty$, which corresponds to always setting one or both variables to 1 or -1, can be dealt with separately via a simple case analysis.

As we discussed in the introduction, a $\mathcal{THRESH}^-$ rounding scheme for MAX DI-CUT is not necessarily odd, but as the following lemma shows, the simple and symmetric structure of our construction ensures that any finite critical point of s is necessarily symmetric around the origin.

Lemma III.4. *Let $x, y \in \mathbb{R}$. If (x,y) is a critical point of $s(t_1,t_2)$, then $x+y=0$ with $y \geq 0$ and $x \leq 0$.*

Proof. Recall that by Proposition II.14

$$\frac{\partial}{\partial t_1}\Phi_\rho(t_1,t_2) \;=\; \varphi(t_1)\cdot\Phi\left(\frac{t_2-\rho t_1}{\sqrt{1-\rho^2}}\right) \ .$$

The partial derivatives of $s(t_1,t_2)$ are

$$\begin{aligned}
\frac{\partial s}{\partial t_1} &= p_1\left(\varphi(t_1) - 2\varphi(t_1)\cdot\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}t_1\right)\right) \\
&\quad + p_2\left(-\varphi(t_1)\cdot\Phi\left(\frac{t_2-\rho_2 t_1}{\sqrt{1-\rho_2^2}}\right)\right), \\
\frac{\partial s}{\partial t_2} &= p_1\left(\varphi(t_2) - 2\varphi(t_2)\cdot\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}t_2\right)\right) \\
&\quad + p_2\left(\varphi(t_2) - \varphi(t_2)\cdot\Phi\left(\frac{t_1-\rho_2 t_2}{\sqrt{1-\rho_2^2}}\right)\right).
\end{aligned}$$

In the above computation, we used Proposition II.14 and the chain rule. Since (x,y) is a critical point of s and φ is strictly positive, we have

$$p_1\left(1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right)\right) + p_2\left(-\Phi\left(\frac{y-\rho_2 x}{\sqrt{1-\rho_2^2}}\right)\right) = 0 \ ,$$

$$p_1\left(1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}y\right)\right) + p_2\left(1-\Phi\left(\frac{x-\rho_2 y}{\sqrt{1-\rho_2^2}}\right)\right) = 0 \ .$$

The first equation can be rewritten as

$$p_1\left(1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right)\right) \;=\; p_2\cdot\Phi\left(\frac{y-\rho_2 x}{\sqrt{1-\rho_2^2}}\right) \ . \quad (1)$$

Since Φ is a positive function, the right hand side of (1) is positive and therefore we have $1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right) > 0$, which implies that $x < 0$.

Since $1-\Phi(t) = \Phi(-t)$, the second equation can be rewritten as

$$p_1\left(1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}y\right)\right) \;=\; -p_2\cdot\Phi\left(\frac{-x+\rho_2 y}{\sqrt{1-\rho_2^2}}\right) \ . \quad (2)$$

By similar logic we can deduce that $y > 0$. We now show that we must have $|x| = |y|$. Assume for the sake of contradiction that $|x| \neq |y|$. We have two cases:

- $|x| > |y|$. It follows that

$$\begin{aligned}
&p_1\cdot\left|1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right)\right| \\
&= p_1\cdot\left|\Phi\left(-\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right) - \Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}x\right)\right| \\
&> p_1\cdot\left|\Phi\left(-\sqrt{\frac{1-\rho_1}{1+\rho_1}}y\right) - \Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}y\right)\right| \\
&= p_1\cdot\left|1-2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}y\right)\right| \ .
\end{aligned}$$

  Note that here we again used $1-\Phi(t) = \Phi(-t)$, as well as the fact that $|\Phi(t) - \Phi(-t)|$ is an increasing function in $|t|$. On the other hand, by (1) and (2) this implies that

$$\left|p_2\cdot\Phi\left(\frac{y-\rho_2 x}{\sqrt{1-\rho_2^2}}\right)\right| \;>\; \left|-p_2\cdot\Phi\left(\frac{-x+\rho_2 y}{\sqrt{1-\rho_2^2}}\right)\right| \ .$$

  Since Φ is a positive and monotone function, this implies that

$$\frac{y-\rho_2 x}{\sqrt{1-\rho_2^2}} \;>\; \frac{-x+\rho_2 y}{\sqrt{1-\rho_2^2}} \ ,$$

  Rearranging the terms, we obtain

$$(1-\rho_2)y \;>\; (1-\rho_2)\cdot(-x) \ .$$

  But this would imply that $|y| > |x|$, which contradicts our assumption.
- $|y| > |x|$. This can be dealt with in a similar manner.

We conclude that we must have then $x + y = 0$ with $y \geq 0$ and $x \leq 0$. $\square$

Lemma III.5. *If $p_1 > p_2$, then $s(t_1, t_2)$ has a unique critical point.*

Proof. Assume (x, y) is a critical point. In the previous lemma, we established that $x = -y < 0$, so we can now plug $y = -x$ into (1) and get

$$\begin{aligned} p_1 \left(1 - 2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}\, x\right)\right) &= p_2\, \Phi\left(\frac{-1-\rho_2}{\sqrt{1-\rho_2^2}}\, x\right) \\ &= p_2\, \Phi\left(-\sqrt{\frac{1+\rho_2}{1-\rho_2}}\, x\right) . \end{aligned}$$

We need to show the equation above has only one solution when $p_1 > p_2$. To this end, define

$$g(t) = p_1 \left(1 - 2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}}\, t\right)\right) - p_2\, \Phi\left(-\sqrt{\frac{1+\rho_2}{1-\rho_2}}\, t\right).$$

We have $g(0) = -p_2/2 < 0$ and $\lim_{t\to-\infty} g(t) = p_1 - p_2 > 0$, so $g(t) = 0$ has at least one solution in $(-\infty, 0)$ by Intermediate Value Theorem. To show that the solution is unique, we compute the derivative of g:

$$\begin{aligned} g'(t) &= p_1 \left(-2\sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot \varphi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot t\right)\right) \\ &\quad + p_2 \cdot \sqrt{\frac{1+\rho_2}{1-\rho_2}} \cdot \varphi\left(-\sqrt{\frac{1+\rho_2}{1-\rho_2}} \cdot t\right) . \end{aligned}$$

By setting $g'(t) = 0$, we obtain

$$\begin{aligned} &2p_1 \cdot \sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot \varphi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot t\right) \\ =& p_2 \cdot \sqrt{\frac{1+\rho_2}{1-\rho_2}} \cdot \varphi\left(-\sqrt{\frac{1+\rho_2}{1-\rho_2}} \cdot t\right). \end{aligned}$$

Plugging in the definition of φ, we get

$$\begin{aligned} &2p_1 \cdot \sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1-\rho_1}{1+\rho_1} \cdot \frac{t^2}{2}\right) \\ =& p_2 \cdot \sqrt{\frac{1+\rho_2}{1-\rho_2}} \cdot \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1+\rho_2}{1-\rho_2} \cdot \frac{t^2}{2}\right), \end{aligned}$$

which is equivalent to

$$\begin{aligned} &2p_1 \cdot \sqrt{\frac{1-\rho_1}{1+\rho_1}} \cdot \exp\left(-\left(\frac{1-\rho_1}{1+\rho_1} - \frac{1+\rho_2}{1-\rho_2}\right) \cdot \frac{t^2}{2}\right) \\ =& p_2 \cdot \sqrt{\frac{1+\rho_2}{1-\rho_2}} . \end{aligned}$$

Since $\rho_1, \rho_2 < 0$ and exp is monotone, this equation has exactly one solution $t^* \in (-\infty, 0)$. Furthermore, $g'(t) > 0$ for $t \in (-\infty, t^*)$ and $g'(t) < 0$ for $t \in (t^*, 0)$. It follows that g has no root in $(-\infty, t^*)$ and has a unique root in $(t^*, 0)$. $\square$

We now deal with the boundary cases. Since our distribution is symmetric with respect to flip, it is sufficient to look at the case where $t_1 = \pm\infty$.

Lemma III.6. *We have $s(+\infty, +\infty) = s(-\infty, -\infty) = s(+\infty, -\infty) = 0$, $s(-\infty, +\infty) = p_2$. For $t_2 \in \mathbb{R}$, we have $s(-\infty, t_2) > s(+\infty, t_2)$. Furthermore, if $p_1 > p_2$, then $s(-\infty, t_2)$ is maximized when $t_2 = t^* = \sqrt{\frac{1+\rho_1}{1-\rho_1}} \cdot \Phi^{-1}(\frac{p_1+p_2}{2p_1})$.*

Proof. Setting a threshold to $+\infty$ corresponds to always setting a variable to false, and $-\infty$ corresponds to always true. When $(t_1, t_2) \in \{(+\infty, +\infty), (-\infty, -\infty), (+\infty, -\infty)\}$, none of the configurations are satisfied, giving a soundness of 0. When $(t_1, t_2) = (-\infty, +\infty)$, only the second configuration is satisfied and this gives a soundness of p_2. For aim, we have

$$\begin{aligned} s(-\infty, t_2) =& p_2 \cdot \Phi(t_2) + p_1 \cdot \Phi_{-\rho_1}(t_2, -t_2) \\ >& p_1 \cdot \Phi_{-\rho_1}(t_2, -t_2) \;=\; s(+\infty, t_2) , \end{aligned}$$

and

$$\frac{\partial s(-\infty, t_2)}{\partial t_2} = \varphi(t_2) \left(p_2 + p_1 \left(1 - 2\Phi\left(\sqrt{\frac{1-\rho_1}{1+\rho_1}} t_2\right)\right)\right) .$$

When $p_1 > p_2$, we have $\frac{\partial s(-\infty, t_2)}{\partial t_2} > 0$ on $(-\infty, t^*)$ and $\frac{\partial s(-\infty, t_2)}{\partial t_2} < 0$ on (t^*, ∞). $\square$

With Lemma III.5 and Lemma III.6, it becomes very easy to determine the maximum of s by simply computing the unique critical point and comparing it with the boundary cases. It turns out that when $b = 0.1757079776$, $c = -0.6876930116$, $p_1 = 0.3770580295$, the unique critical point of $s(t_1, t_2)$ is at $(-t_0, t_0)$ where $t_0 \simeq 0.1887837358$, which is also a global maximum whose value is about 0.8746024732. A plot of $s(t_1, t_2)/\mathsf{Comp}(\tilde{\Theta})$ with these parameters can be found in Figure 1. It follows that with these parameters, any $\mathcal{THRESH}^-$ rounding scheme achieves a ratio of at most 0.87461. This can then be converted into Unique Games hardness, with the help of the following theorem.

Theorem III.7 ([24]). *Let P be a predicate of arity 2. Let $\tilde{\Theta}$ be a distribution of positive configurations for* MAX CSP$^+(P)$ *such that $\mathsf{Comp}(\tilde{\Theta}) = c$. If $\mathsf{Sound}(\tilde{\Theta}, f) \leq s$ for every $f : [-1, 1] \to \mathbb{R}$, then it is UG-hard to approximate* MAX CSP$^+(P)$ *within a ratio of $s/c + \epsilon$ for any $\varepsilon > 0$.*

The above theorem was originally stated in terms of MAX CSP(P) in [24], but the proof is essentially the same for MAX CSP$^+(P)$. We include a proof in the appendix of the full version for completeness.

B. *Intuition for the upper bound*

While we found this integrality gap instance with a computer search, we now give some intuition for why this integrality gap instance works well. Previously, the best algorithm for

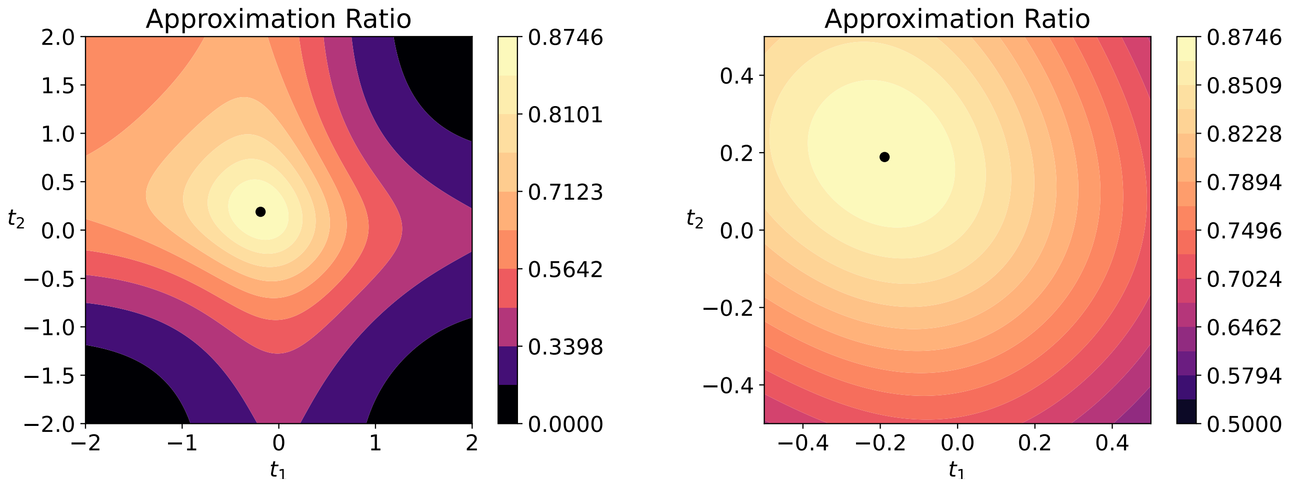


Fig. 1. Contour plots of $s(t_1, t_2)/\mathsf{Comp}(\tilde{\Theta})$ with optimal parameters. The black dot represents the global maximum $(-t_0, t_0)$ where $t_0 \simeq 0.1887837358$. All plots in this paper are made with Matplotlib [30].

MAX DI-CUT was the LLZ algorithm [4] which works equally well for MAX 2-AND.

If we restrict our attention to points $(b_1, b_2, -1 + |b_1 + b_2|)$ where the triangle inequality is tight (so the completeness is as large as possible given b_1 and b_2), using experimental simulations, the performance of LLZ in terms of b_1 and b_2 is shown in Figure 2.

We observe that there is a strip where $b_1 + b_2 \approx .35$ and a strip where $b_1 + b_2 \approx -.35$ where the LLZ algorithm does poorly. In order to reduce the degrees of freedom for rounding schemes for our instance, it makes sense to choose $b_2 = b_1 = \pm b$. With this choice, there are only two degrees of freedom, the threshold for b and the threshold for $-b$. $b \simeq 0.1757079776$ puts us right in the middle of the hard strips for LLZ.

Once we have these two points, we can also add points of the form $(b, -b, c)$ and $(-b, b, c')$ without additional degrees of freedom. While we originally thought that points where the triangle inequality is tight may be optimal, this turned out to not be the case. Instead, we found experimentally that adding the point $(b, -b, c)$ with $c \simeq -0.6876930116$ worked best. The completeness for $(-b, b, c')$ is too low, so adding this kind of point does not help.

C. Possibly improved upper bounds

We believe that slightly improved upper bounds for MAX DI-CUT can be obtained using more than one pair of biases. In the full version, we give more complicated distributions that use up to 4 pairs of biases that seem to indicate that $\alpha_{\text{DI-CUT}} \leq 0.8745794663$ (not verified rigorously). It would probably be very hard to prove this inequality analytically. It is probably possible to prove that, say, $\alpha_{\text{DI-CUT}} \leq 0.8745795$, using interval arithmetic, but we have not done so yet.

IV. A NEW APPROXIMATION ALGORITHM FOR MAX DI-CUT

In this section, we present the techniques used for proving Theorem I.3. We first briefly give some intuition for why a rounding scheme for MAX DI-CUT better than those possible for MAX 2-AND should exist. Then, after describing the rounding scheme, we explain how this rounding scheme was discovered experimentally. Finally, we discuss how we rigorously verify the approximation guarantees of this rounding scheme using interval arithmetic.

A. Intuition for the separation between MAX 2-AND and MAX DI-CUT

We now try to give some intuition for why there is a gap between MAX 2-AND and MAX DI-CUT. We first observe that Austrin's hard distributions of configurations for MAX 2-AND (see Section 6 of [24]) can be easily beaten for MAX DI-CUT. For simplicity, we consider Austrin's simpler two-configuration distribution which is as follows

1) $(0, -b, b-1)$ with probability 0.64612
2) $(0, b, b-1)$ with probability 0.35388

where $b = 0.33633$. This gives an inapproximability of 0.87451 for MAX 2-AND.

For MAX 2-AND, since variables can be negated, we can assume without loss of generality that when $b = 0$, for each rounding scheme in our distribution the variable has an equal probability of being rounded to true or false.

For MAX DI-CUT, we only have the symmetry of $\theta \mapsto \text{flip}(\theta)$. When we add this symmetry to the integrality gap instance, we obtain:

1) $(0, -b, b-1)$ with probability 0.32306
2) $(b, 0, b-1)$ with probability 0.32306
3) $(0, b, b-1)$ with probability 0.17694

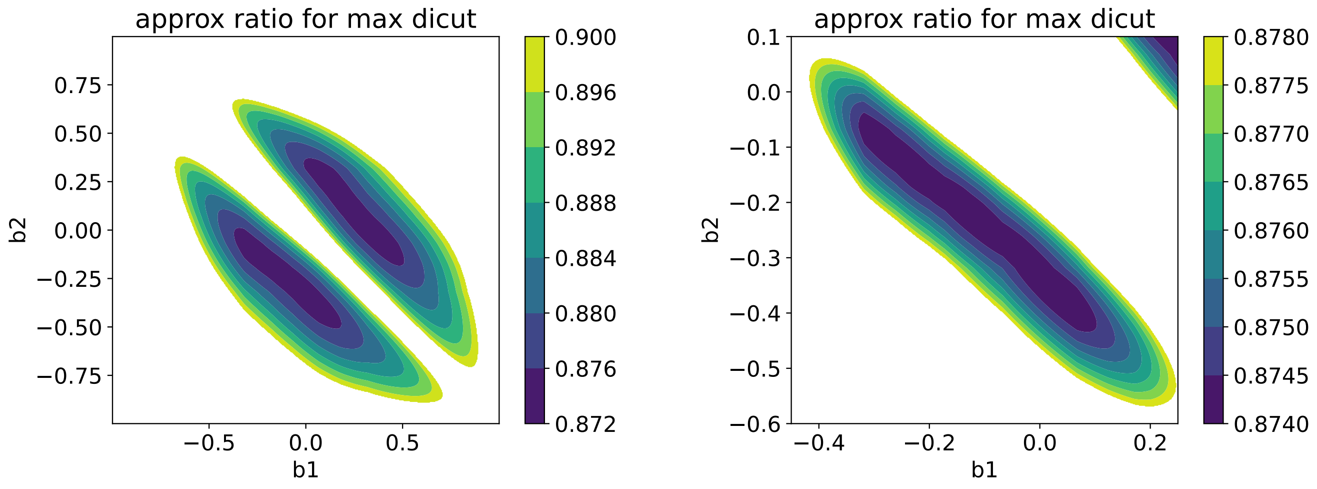


Fig. 2. Contour plots of the performance of the LLZ function [4] for MAX 2-AND and MAX DI-CUT.

4) $(-b, 0, b-1)$ with probability 0.17694

where $b = 0.33633$.

The following distribution of rounding functions trivially satisfies $\frac{1}{2}$ of the configurations of this MAX DI-CUT instance.

1) With probability $\frac{1}{2}$, round all variables with bias 0 to 1 and round all variables with bias $-b$ or b to -1.
2) With probability $\frac{1}{2}$, round all variables with bias 0 to -1 and round all variables with bias $-b$ or b to 1.

Since the completeness of these configurations are all at most $\frac{1}{2}$, we obtain a ratio which is at least 1.

While this is an extreme example, this shows that making the variables with zero or low bias more likely to be rounded to 1 or more likely to be rounded to -1 can help round other variables more effectively as the behavior of these variables is more predictable.

This means that configurations where one or more variables have bias 0 are easier for MAX DI-CUT. Instead, the configurations in our simple distribution (i.e., $(b, b, -1+2b)$ where $b \simeq 0.1757079776$) are hard configurations and all rounding schemes in the distribution have essentially the same behavior at these configurations.

B. The rounding scheme

We now describe a $\mathcal{THRESH}$ scheme, that separates MAX DI-CUT from MAX 2-AND. As mentioned, a $\mathcal{THRESH}$ scheme is a distribution over $\mathcal{THRESH}^-$ schemes. We use discrete distributions over a relatively small number of $\mathcal{THRESH}^-$ schemes. For computational convenience, we choose the $\mathcal{THRESH}^-$ functions to be *piecewise linear* functions. More precisely, to build a piecewise linear function f, we pick a finite set $S \subset [-1, 1]$ of *control points* with $-1, 1 \in S$. For each of these control points $s \in R$, we assign a real threshold $f(s)$ such that the breakpoints of the graph of f are at $(s, f(s))$ for $s \in S$. Then, for every $x \in (-1, 1) \setminus S$, we identify $x_- = \max(S \cap [-1, x))$ and $x_+ = \min(S \cap (x, 1])$, and set

$$f(x) \ = \ f(x_-) + \frac{x - x_-}{x_+ - x_-}(f(x_+) - f(x_-)) \ .$$

We use the same set of control points for every function in our $\mathcal{THRESH}$ scheme.

For our application to MAX DI-CUT, we picked a set S of 17 control points: 0, ± 0.1, ± 0.164720, ± 0.179515, ± 0.25, ± 0.3, ± 0.45, ± 0.7, ± 1. The choice of most control points is fairly arbitrary. It seemed important, however, to choose the four control points ± 0.164720 and ± 0.179515 as they seem to be situated in regions in which very fine control over the values of the rounding functions is needed. Further small improvements are probably possible by slightly moving some of the control points or by adding new control points.

Then, using the algorithm presented in Section IV-C, we produced a "raw" $\mathcal{THRESH}$ rounding scheme which is a probability distribution over 39 piecewise-linear rounding functions. After a careful ad-hoc analysis, we were able to simplify the distribution to a "clean" $\mathcal{THRESH}$ scheme with only 7 piecewise rounding functions, which we summarize in Table I and Figure 3.

It is interesting to note that the function f_1, which is used in about 99.7% of the time, is very close to the single function used by Lewin, Livnat and Zwick [4]. We do not yet have a satisfactory explanation of the shape of the other functions. Some of the values of the functions, especially at control points ± 0.45, ± 0.7 and ± 1 can be changed slightly without affecting the performance ratio obtained. We also note that the last two functions do not seem to contribute much. We have a scheme with only 5 functions with only a very slightly smaller performance ratio.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
prob	0.996902	0.000956	0.000956	0.000393	0.000393	0.000200	0.000200
-1.000000	-1.601709	-2.000000	-2.000000	-0.034381	-0.430994	-2.000000	2.000000
-0.700000	-0.853605	-2.000000	-2.000000	-0.034381	-0.430994	-2.000000	2.000000
-0.450000	-0.517014	-2.000000	-0.629564	-0.440988	-0.896878	-2.000000	2.000000
-0.300000	-0.333109	-1.520523	1.711824	-1.406591	1.643936	-2.070000	1.970000
-0.250000	-0.274589	-0.687582	2.019266	-0.622399	-0.127984	-1.629055	2.070000
-0.179515	-0.192926	-0.195474	-0.229007	-0.268471	-0.339566	-0.544957	-0.103307
-0.164720	-0.175942	-0.381789	-0.649998	-0.116530	-0.073069	-0.361234	-0.575047
-0.100000	-0.105428	-0.026636	-1.175439	0.066139	-0.123693	2.070000	-1.351740
0.000000	0.000000	2.046025	-2.046025	1.728858	-1.728858	2.050000	-2.050000
0.100000	0.105428	1.175439	0.026636	0.123693	-0.066139	1.351740	-2.070000
0.164720	0.175942	0.649998	0.381789	0.073069	0.116530	0.575047	0.361234
0.179515	0.192926	0.229007	0.195474	0.339566	0.268471	0.103307	0.544957
0.250000	0.274589	-2.019266	0.687582	0.127984	0.622399	-2.070000	1.629055
0.300000	0.333109	-1.711824	1.520523	-1.643936	1.406591	-1.970000	2.070000
0.450000	0.517014	0.629564	2.000000	0.896878	0.440988	-2.000000	2.000000
0.700000	0.853605	2.000000	2.000000	0.430994	0.034381	-2.000000	2.000000
1.000000	1.601709	2.000000	2.000000	0.430994	0.034381	-2.000000	2.000000

TABLE I

A $\mathcal{THRESH}$ ROUNDING SCHEME THAT GIVES A RIGOROUSLY VERIFIED APPROXIMATION RATIO OF AT LEAST 0.874473 FOR MAX DI-CUT. (THE ACTUAL RATIO IS PROBABLY ABOUT 0.874502.) THE SCHEME USES 7 PIECEWISE-LINEAR ROUNDING FUNCTIONS $f_1, f_2, \ldots, f_7$ DEFINED ON 17 CONTROL POINTS. THE FUNCTION f_1 IS ODD AND IS VERY CLOSE TO THE SINGLE FUNCTION USED BY LEWIN, LIVNAT AND ZWICK [4]. THE OTHER SIX FUNCTIONS COME IN PAIRS. THE TWO FUNCTIONS IN EACH PAIR ARE FLIPS OF EACH OTHER.

C. Discovery of the $\mathcal{THRESH}$ scheme

In this section, we discuss the process of experimentally discovering the "raw" $\mathcal{THRESH}$ scheme described in Section IV-B. For now, we will make a couple of assumptions, which will be fully worked out in Section IV-C4.

(1) Instead of optimizing over all valid configurations of MAX DI-CUT, we restrict to optimizing over a finite set Θ of configurations, where $\mathsf{Comp}(\theta) > 0$ for all $\theta \in \Theta$.
(2) Let $\mathcal{F}$ be a restricted family of $\mathcal{THRESH}^-$ schemes (e.g., the piecewise linear functions). We shall further assume throughout this discussion that we have access to an oracle $\mathcal{O}_\mathcal{F}$ which, when given a probability distribution $\tilde{\Theta} \in \mathrm{Dist}(\Theta)$, identifies a function $f \in \mathcal{F}$ which maximizes $\mathsf{Sound}(\tilde{\Theta}, f)$.

1) Finite F: a game-theoretic approach

Assume further we have found a finite set $F \subset \mathcal{F}$ of candidate rounding functions. We would like to identify the following:

(a) An optimal (worst) distribution $\tilde{\Theta}$ over Θ such that

$$\tilde{\Theta} \;=\; \operatorname*{argmin}_{\tilde{\Theta}\in\mathrm{Dist}(\Theta)} \; \max_{f\in F} \; \frac{\mathsf{Sound}(\tilde{\Theta}, f)}{\mathsf{Comp}(\tilde{\Theta})} \,.$$

(b) An optimal (best) distribution $\tilde{F}$ over F such that

$$\tilde{F} \;=\; \operatorname*{argmax}_{\tilde{F}\in\mathrm{Dist}(F)} \; \min_{\theta\in\Theta} \; \mathop{\mathbb{E}}_{f\sim\tilde{F}}\left[\frac{\mathsf{Sound}(\tilde{\Theta}, f)}{\mathsf{Comp}(\tilde{\Theta})}\right] .$$

It turns out that both of these objectives can be solved by mutually dual LPs. This is best seen by casting both questions as a *zero-sum game*. Fix a real number α, which should be thought of as an estimate of the approximation ratio of this restricted MAX DI-CUT problem. In our game, which we call *the α-game*, there are two players Alice and Bob that play simultaneously: Alice picks $\theta \in \Theta$ and Bob picks $f \in F$. We then have the following payoffs

$$\begin{aligned} \text{Alice:}\;\; \mathsf{alice}_\alpha(\theta, f) \;&:=\; \alpha\,\mathsf{Comp}(\theta) - \mathsf{Sound}(\theta, f)\\ \text{Bob:}\;\; \mathsf{bob}_\alpha(\theta, f) \;&:=\; \mathsf{Sound}(\theta, f) - \alpha\,\mathsf{Comp}(\theta) \end{aligned}$$

Note that this game is a finite zero-sum game and thus by standard theory (e.g., Von Neumann's minimax theorem [31] and Nash equilibria [32]), there is a single[5] Nash-equilibrium $(\tilde{\Theta}_\alpha, \tilde{F}_\alpha)$ which is the optimal mixed strategy for both players. Let $v(\alpha)$ be the expected payoff of this optimal strategy for Alice (i.e., the *value* of the game). We now make the following simple observation.

Proposition IV.1. *The function $v(\alpha)$ is strictly increasing in α.*

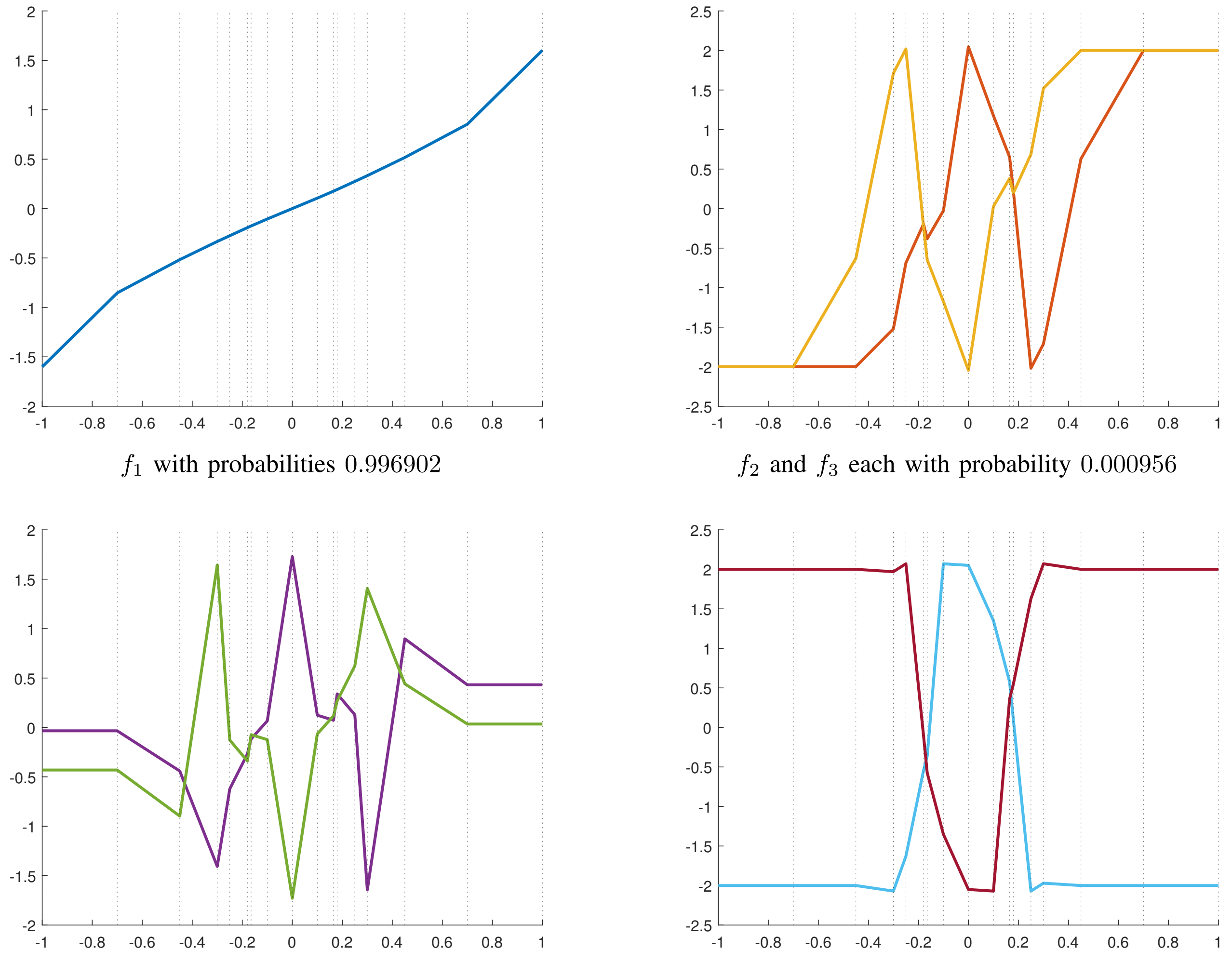
Proof. Fix $\alpha < \alpha'$. Assume for the α'-game that Alice plays $\tilde{\Theta}_\alpha$. Assume Bob plays an arbitrary mixed strategy $\tilde{F}$. Then, Alice's expected payoff is

$$\begin{aligned} &\mathop{\mathbb{E}}_{\theta\sim\tilde{\Theta}_\alpha, f\sim\tilde{F}}[\mathsf{alice}_{\alpha'}(\theta, f)]\\ &= \mathop{\mathbb{E}}_{\theta\sim\tilde{\Theta}_\alpha, f\sim\tilde{F}}[\mathsf{alice}_\alpha(\theta, f) + (\alpha'-\alpha)\mathsf{Comp}(\theta)]\\ &= \mathop{\mathbb{E}}_{\theta\sim\tilde{\Theta}_\alpha, f\sim\tilde{F}}[\mathsf{alice}_\alpha(\theta, f)] + (\alpha'-\alpha)\mathop{\mathbb{E}}_{\theta\sim\tilde{\Theta}_\alpha}[\mathsf{Comp}(\theta)]\\ &> v(\alpha)\,, \end{aligned}$$

where we use the fact that $\tilde{\Theta}_\alpha$ is the Nash equilibrium for the α-game and that $\mathsf{Comp}(\theta) > 0$ for all $\theta \in \Theta$. In other words, Alice can assure for the α'-game a payoff strictly greater than $v(\alpha)$. Thus, $v(\alpha') > v(\alpha)$. □

It is easy to see that $v(0) \le 0$ (as $\mathsf{alice}_0 \le 0$). Further $\mathsf{alice}_\alpha(\theta, f) \to \infty$ as $\alpha \to \infty$. Thus, by Proposition IV.1, there

[5]Depending on the singular values of the payoff matrices, there may be multiple Nash-equilibrium, but they all have the same value. In that situation, we pick one of the Nash equilibriums arbitrarily to be representative Nash equilibrium.



f_1 with probabilities 0.996902

f_2 and f_3 each with probability 0.000956

f_4 and f_5 each with probability 0.000393

f_6 and f_7 each with probability 0.000200

Fig. 3. Plots of the seven rounding functions used in the $\mathcal{THRESH}$ rounding scheme given in Table I that achieves a verified approximation ration of at least 0.87446 for MAX DI-CUT.

is a unique $\alpha_{\Theta,F}$ for which $v(\alpha_{\Theta,F}) = 0$ with a corresponding Nash equilibrium of $\tilde{\Theta}_F$ and $\tilde{F}_\Theta$. Unpacking the definition of Nash equilibrium and using the fact that $\mathsf{Sound}(\tilde{\Theta}, f)$ is an affine function in $\tilde{\Theta}$, we have that

(a) For all $f \in F$, we have that

$$\frac{\mathsf{Sound}(\tilde{\Theta}_F, f)}{\mathsf{Comp}(\tilde{\Theta})} \le \alpha\ .$$

(b) For all $\theta \in \Theta$, we have that

$$\mathop{\mathbb{E}}_{f\sim\tilde{F}_\Theta}\left[\frac{\mathsf{Sound}(\theta, f)}{\mathsf{Comp}(\theta)}\right] \ge \alpha\ .$$

Thus, $\tilde{\Theta}_F$ and $\tilde{F}_\Theta$ are the optimal distributions for problems (a) and (b) from before. We can efficiently compute these distributions through a suitable linear program. Let w_θ be the weights of the optimal distribution $\tilde{\Theta}_F$ and let p_f be the weights of the optimal distribution $\tilde{F}_\Theta$. By definition of the Nash equilibrium, we have that

$$\sum_{\theta\in\Theta} w_\theta(\alpha_{\Theta,F}\,\mathsf{Comp}(\theta) - \mathsf{Sound}(\theta, f)) \ge 0 \quad , \quad \forall f \in F \tag{3}$$

$$\sum_{f\in F} p_f(\mathsf{Sound}(\theta, f) - \alpha_{\Theta,F}\,\mathsf{Comp}(\theta)) \ge 0 \quad , \quad \forall \theta \in \Theta \tag{4}$$

To formulate this as a pair of linear programs, we will have $\alpha_{\Theta,F}$ be our objective. Since $v(\alpha) \ge 0$ for all $\alpha \ge \alpha_{\Theta,F}$ we will have a "minimize" objective to compute the w_θ's and a "maximize" objective to compute the p_f's.

However, neither set of constraints is currently an LP as $\alpha_{\Theta,F}$ is also a variable of our LP (in fact the objective function). This is easy to fix for (4), as $\sum_{f\in F} p_f = 1$, so

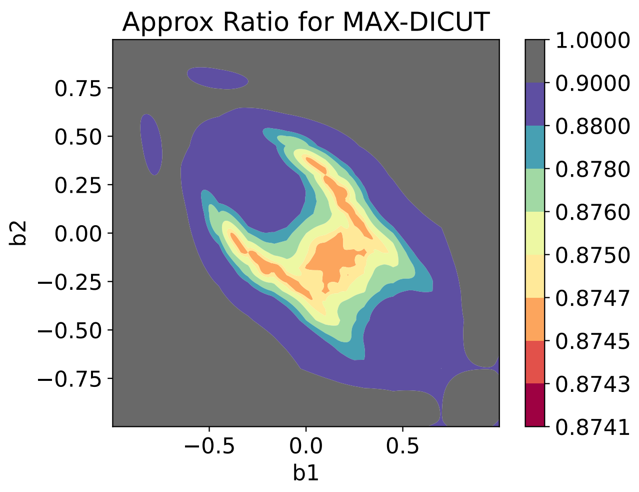


Fig. 4. This plot is a contour plot of the performance of the $\mathcal{THRESH}$ scheme with 7 piecewise-linear rounding functions for various choices of b_1 and b_2 (with an approximately worst-case choice of b_{12}) selected.

we can rewrite (4) as

$$\sum_{f\in F} p_f \mathsf{Sound}(\theta, f) \;\geq\; \alpha_{\Theta,F}\,\mathsf{Comp}(\theta)\ .$$

For (3), we use a 'clever' trick. We renormalize the weights so that $\sum_\theta \hat{w}_\theta \mathsf{Comp}(\theta) = 1$ instead of $\sum_\theta w_\theta = 1$, and use the $\hat{w}_\theta$'s as the variables of the LP. With this normalization, we then get the linear constraints

$$\sum_{\theta\in\Theta} \hat{w}_\theta \mathsf{Sound}(\theta, f) \;\leq\; \alpha_{\Theta,F}\ .$$

After solving the LP, We can find the original weights by setting $w_\theta = \hat{w}_\theta / \sum_{\theta'\in\Theta} \hat{w}_{\theta'}$. Formally, the two LPs we solve are as follows.

Primal LP (finding $\tilde{\Theta}_F$)

minimize: α

subject to: $\sum_{\theta\in\Theta} \mathsf{Sound}(\theta_i, f)\hat{w}_\theta \leq \alpha \quad , \quad \forall f \in F$

$\sum_{\theta\in\Theta} \mathsf{Comp}(\theta_i)\hat{w}_\theta = 1$

$\hat{w}_\theta \geq 0 \quad , \quad \forall \theta \in \Theta$

Dual LP (finding $\tilde{F}_\Theta$)

maximize: α

subject to: $\sum_{f\in F} \mathsf{Sound}(\theta, f_j)p_f$

$\geq \alpha\,\mathsf{Comp}(\theta) \quad , \quad \forall \theta \in \Theta$

$\sum_{f\in F} p_f = 1$

$p_f \geq 0 \quad , \quad \forall f \in F$

It is straightforward to prove that these two LPs are dual to each other and thus will both achieve the same objective value $\alpha_{\Theta,F}$

2) Extending to infinite $\mathcal{F}$

Since the full family of functions we optimize over is infinite, we cannot hope to find a (near) optimal distribution over $\mathcal{F}$ by just solving a suitable finite linear program. Instead, we work with a small set of candidate functions F which we iteratively improve. In particular, for a fixed F, we can compute the hardest distribution $\tilde{\Theta}_F$ for this family of functions and then use the oracle to find the function $f = \mathcal{O}_\mathcal{F}(\tilde{\Theta}_F)$ which does the best on this hard distribution. (Note that $\mathcal{O}_\mathcal{F}(\tilde{\Theta}_F)$ needs to solve a non-linear, and probably non-convex, optimization problem.) We add f to F and continue for some fixed number T of steps. We note that similar minimax algorithms are prevalent in machine learning, such as in generative adversarial networks [33]. See Algorithm 1 for the formal details.

Algorithm 1 $\mathcal{THRESH}$ discovery algorithm (fixed Θ)

```
 1: procedure FINDTHRESH($\Theta$, $T$)
 2:     Pick an initial distribution $\tilde{\Theta}_0 \in \mathrm{Dist}(\Theta)$
 3:     $f_1 \leftarrow \mathcal{O}_\mathcal{F}(\tilde{\Theta}_0)$
 4:     $F_1 \leftarrow \{f_1\}$.
 5:     for $i \in \{1, 2, \ldots, T-1\}$ do
 6:         Find the hardest $\tilde{\Theta}_i$ for $F_i$ using the Primal LP
            with objective value $\alpha_i$
 7:         $f_{i+1} \leftarrow \mathcal{O}_\mathcal{F}(\tilde{\Theta}_i)$
 8:         $F_{i+1} \leftarrow F_i \cup \{f_{i+1}\}$
 9:     end for
10:     Find the optimal distribution $\tilde{F}_T$ over $F_T$ for $\Theta$ using
        the Dual LP
11:     return $\tilde{F}_T$
12: end procedure
```

It is easy to see that the objective value α_i of the Primal LP in Algorithm 1 increases at each step of the loop. Further, it is not hard to prove that $\alpha_i \leq 1/\min_{\theta\in\Theta} \mathsf{Comp}(\theta)$. Thus, as $T \to \infty$, the objective value of the Primal LP tends to a limit $\alpha_{\lim}$. Our main correctness guarantee of our algorithm is that we converge to this limit at an effective rate and that this limit is the best we can hope for.

Theorem IV.2. *Fix $\varepsilon > 0$ and assume $C := 1/\min_{\theta\in\Theta} \mathsf{Comp}(\theta)$ is finite. Let α_T be the objective value of the Dual LP computing $\tilde{F}_T$. Assume that $T > (C/\varepsilon)^{|\Theta|}$, then $\alpha_T \geq \alpha_{\lim} - \varepsilon$. Further, for every finite distribution $\tilde{F}$ over functions in $\mathcal{F}$,*

$$\mathop{\mathbb{E}}_{f\sim\tilde{F}}\left[\frac{\mathsf{Sound}(\tilde{\Theta}, f)}{\mathsf{Comp}(\tilde{\Theta})}\right] \;\leq\; \alpha_{\lim}\ .$$

Proof. Observe that for all $j > i \geq 1$, we have that

$$\frac{\mathsf{Sound}(\tilde{\Theta}_i, f_{i+1})}{\mathsf{Comp}(\tilde{\Theta}_i)} \;\geq\; \frac{\mathsf{Sound}(\tilde{\Theta}_j, f_j)}{\mathsf{Comp}(\tilde{\Theta}_j)}\ ,$$

because the distribution $\tilde{\Theta}_i$ certifies that no finite distribution of rounding functions over F_j can do better than $\frac{\mathsf{Sound}(\tilde{\Theta}_i,\mathcal{O}_{\mathcal{F}}(\tilde{\Theta}_i))}{\mathsf{Comp}(\tilde{\Theta}_i)}$. In particular, by taking the limit as $j \to \infty$, this implies that for all $i \ge 1$,

$$\frac{\mathsf{Sound}(\tilde{\Theta}_i, f_{i+1})}{\mathsf{Comp}(\tilde{\Theta}_i)} \;\ge\; \alpha_{\text{lim}}\ . \tag{5}$$

Assume for sake of contradiction that $\alpha_T < \alpha_{\text{lim}} - \varepsilon$. Thus, $\alpha_i < \alpha_{\text{lim}} - \varepsilon$ for all $i \in \{1, \dots, T-1\}$. Pick $\delta = \varepsilon/C$. Define the function $s_\Theta : \mathcal{F} \to [0,1]^\Theta$ as

$$s_\Theta(f) \;=\; (\mathsf{Sound}(\theta, f) : \theta \in \Theta)\ .$$

Observe that if f and f' are such that $\|s_\Theta(f) - s_\Theta(f')\|_\infty \le \delta$, then for any fixed distribution $\tilde{\Theta}$, we have that

$$\left|\frac{\mathsf{Sound}(\tilde{\Theta}, f)}{\mathsf{Comp}(\tilde{\Theta})} - \frac{\mathsf{Sound}(\tilde{\Theta}, f')}{\mathsf{Comp}(\tilde{\Theta})}\right| \;\le\; \frac{\delta}{\min_{\theta\in\Theta}\mathsf{Comp}(\theta)} \;=\; \varepsilon\ . \tag{6}$$

Divide $[0,1]^\Theta$ in $(1/\delta)^{|\Theta|}$ hypercubes with ℓ_∞-diameter δ. Let $\mathcal{H}$ be the this family of hypercubes. Since $T > (C/\varepsilon)^{|\Theta|}$, by the pigeonhole principle there exists $i, j \in \{1, \dots, T\}$ with f_i and f_j in the same hypercube but $i < j$. In particular, we have by the minimax guarantee of Θ_{j-1}, (6), and (5),

$$\begin{aligned}\alpha_T \;\ge\; \alpha_{j-1} \;&\ge\; \frac{\mathsf{Sound}(\tilde{\Theta}_{j-1}, f_i)}{\mathsf{Comp}(\tilde{\Theta}_{j-1})}\\ &\ge\; \frac{\mathsf{Sound}(\tilde{\Theta}_{j-1}, f_j)}{\mathsf{Comp}(\tilde{\Theta}_{j-1})} - \varepsilon \;\ge\; \alpha_{\text{lim}} - \varepsilon\ ,\end{aligned} \tag{7}$$

as desired.

For the claim about $\tilde{F}$, assume for sake of contradiction that there is a $\varepsilon' > 0$ such that

$$\mathop{\mathbb{E}}_{f\sim\tilde{F}}\left[\frac{\mathsf{Sound}(\tilde{\Theta}, f)}{\mathsf{Comp}(\tilde{\Theta})}\right] \;\ge\; \alpha_{\text{lim}} + \varepsilon'\ .$$

Then, we must have that for all $i \ge 1$,

$$\frac{\mathsf{Sound}(\tilde{\Theta}_i, f_{i+1})}{\mathsf{Comp}(\tilde{\Theta}_i)} \;\ge\; \alpha_{\text{lim}} + \varepsilon'\ .$$

However, if we take the limit in (7) as $\varepsilon \to 0$ and $T \ge (C/\varepsilon)^{|\Theta|}$, we obtain that $\alpha \ge \alpha_{\text{lim}} + \varepsilon'$, a contradiction. □

Remark IV.3. *The second claim of* Theorem IV.2 *can also be proved for continuous distributions $\tilde{\mathcal{F}}$ over $\mathcal{F}$. In that case, we can approximately discretize $\tilde{\mathcal{F}}$ by picking representative functions which cover the space of functions in the ℓ_∞ metric with respect to s_Θ. We omit further details.*

Remark IV.4. *Although this proof only gives correctness when T is exponential in the size of Θ, in practice our simulation only requires $T = |\Theta|^{O(1)}$ rounds to converge with $\varepsilon \approx 10^{-6}$. Perhaps this suggests that the theoretical analysis can also be improved.*

3) *Extension to infinite Θ*

We now briefly discuss how to extend Algorithm 1 to allow Θ to grow. Let $\Theta_{\text{valid},\varepsilon}$ be the space of all valid configurations of MAX DI-CUT with completeness at least ε. Assume we also have an oracle $\mathcal{O}_\Theta$ which when given a distribution of rounding functions $\tilde{F}$ outputs the configuration $\theta \in \Theta_{\text{valid},\varepsilon}$ on which $\tilde{F}$ performs the worst. We can then dynamically grow our "working set" of configurations Θ using the following procedure.

Algorithm 2 $\mathcal{THRESH}$ discovery algorithm (growing Θ)

```
1: procedure FINDTHRESHFULL($T$, $T'$)
2:     Pick $\Theta_0$ arbitrarily.
3:     for $i \in \{1, 2, \dots, T'-1\}$ do
4:         $\tilde{F}_i \leftarrow \mathsf{FindThresh}(\Theta_{i-1}, T)$.
5:         $\theta_i \leftarrow \mathcal{O}_\Theta(\tilde{F}_i)$
6:         $\Theta_i \leftarrow \Theta_{i-1} \cup \{\theta_i\}$.
7:     end for
8:     return $\mathsf{FindThresh}(\Theta_{T'}, T)$
9: end procedure
```

Let α_i the performance guarantee of $\tilde{F}_i$ over Θ_{i-1} and let $\hat{\alpha}_i$ be optimal approximation ratio if T were to tend to ∞. Note that $\hat{\alpha}_i$ must monotonically decrease (although non-necessarily strictly). Since each $\hat{\alpha}_i$ is nonnegative, they must have a limit $\hat{\alpha}_{\text{lim}}$. Via an argument similar[6] to Theorem IV.2, we can take an δ-net over the configuration space $\Theta_{\text{valid},\varepsilon}$ and argue that if both θ_i and θ_j are in the same region of the δ-net, then $\tilde{F}_j$ must perform with a ratio at least $\hat{\alpha}_{\text{lim}} - \varepsilon$ on all configurations[7] in $\Theta_{\text{valid},\varepsilon}$. In particular, we can guarantee that when T' is sufficiently large, then nearly all $\tilde{F}_i$'s with $i \in \{T'/2, \dots, T'\}$ are near-optimal distributions. This proves to be an adequate guarantee for practical simulation.

4) *Implementation details*

We now discuss the implementation details for how the "raw" $\mathcal{THRESH}$ scheme was generated as well as details of how the "clean" $\mathcal{THRESH}$ scheme was derived from it.

a) The raw distribution.

Overall, the algorithm for discovering the "raw" $\mathcal{THRESH}$ distribution was implemented in Python (version 3.10).

The oracle $\mathcal{O}_{\mathcal{F}}$ is computed using the SciPy library's minimize routine [34] which finds a locally maximal rounding function f when given a starting function $\hat{f} : S \to (-\infty, \infty)$ as input. For numerical stability, we assume that all thresholds are in the range $[-2, 2]$. We compute $\mathcal{O}_\Theta$ by computing $\mathsf{Sound}(\theta, \tilde{F})$ for θ's in a suitably spaced grid and then calling minimize on the worst grid point to further tune the parameters.

[6]This further requires that the family of functions $\mathcal{F}$ is uniformly continuous: that is small changes in the configurations imply that the rounding functions do not change much. This is true for uniformly bounded, piecewise linear functions.

[7]In practice, the distribution of functions also does well on instances with completeness less than ε.

The Sound routine was computed using Genz's numerical algorithms for approximate multivariate normal integration [35], [36] which is bundled with SciPy. The linear programming routines were implemented using CVXPY [37], [38] as a wrapper around the ECOS solver [39].

In practice, we found that the convergence was more stable by additionally adding $\mathrm{flip}(f_i)$ to F_i in Algorithm 1. Likewise, in Algorithm 2, it was best to add $\mathrm{flip}(\theta_i)$ along with θ_i.

Perhaps the most sensitive part of this algorithm is the choice of the initial Θ_0 in Algorithm 2. We found it best to set Θ_0 to be a near-optimal hard distribution. With this choice, it only took $T' \approx 150$. In practice, we did not aim for a fixed T in Algorithm 1, but rather a more complicated stopping criteria based on how fast α_i is stabilizing. This roughly translates to $T \ll 100$. In total, it took a few hours of single-core computation on a standard desktop computer to find the $\mathcal{THRESH}$ function described in Section IV-B. However, as mentioned in Section IV-C3, the worst-case performance of the distribution $\tilde{F}_i$ is not monotone in i, so it took a few instances of trial and error (i.e., run for a few more iterations) until the worst-case performance was satisfactory.

We further remark that routines similar to the ones described in this section were used to discover (approximately) the configurations used to prove the upper bound on MAX DI-CUT in Section III (in this case Θ_0 was seeded to be a fixed ε-spaced grid).

b) The clean distribution.

Inspecting the 39 functions of the raw distribution revealed that they naturally divide into 7 families of functions, with the functions in each family being fairly similar to each other. Taking a weighted average of the functions in each family yielded a scheme with only 7 functions that did almost as well as the original scheme. Further inspection revealed that one of these 7 functions was almost odd, and that the other six functions divide into three pairs in which functions are close to being flips of each other. The first function was made odd by taking the average of the function and its flip. Similarly, the functions in each pair were made flips of each other. This slightly improved the performance ratio obtained. Finally, numerical optimization was used to perform small optimizations. The resulting 7 functions are the ones given in Table I. The final performance ratio obtained was slightly better than the one achieved by the raw distribution. The computations and optimizations were done using MATLAB.

D. Verification using interval arithmetic

1) Sketch of the algorithm

From now on, we use $\tilde{F}$ to refer to the clean distribution of 7 functions from the previous subsection. To prove that the claimed distribution $\tilde{F}$ of rounding functions achieves an approximation ratio of at least α for MAX DI-CUT, we need to show that

$$\forall\theta, \mathsf{Comp}(\theta) \neq 0 \implies \frac{\mathbb{E}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)]}{\mathsf{Comp}(\theta)} \geq \alpha,$$

or equivalently

$$\forall\theta, \quad \mathop{\mathbb{E}}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha\cdot\mathsf{Comp}(\theta) \ \geq\ 0\ .$$

Note that in the above expression, $\mathsf{Comp}(\theta)$ only involves simple arithmetic operations, and $\mathbb{E}[\mathsf{Sound}(\theta, f)]$ is a weighted sum of two-dimensional Gaussian integrals, while θ takes value in $[-1, 1]^3$, modulo the triangle inequalities.

To rigorously verify the inequality for all configurations, we deploy the technique of *interval arithmetic*. In interval arithmetic, instead of doing arithmetics with numbers, we apply arithmetic operations to intervals. Let op be a k-ary operation and $I_1, I_2, \ldots, I_k$ be k intervals, then the interval arithmetic on $op(I_1, I_2, \ldots, I_k)$ will produce an interval I_{op} with the following *rigorous* guarantee: $op(x_1, x_2, \ldots, x_k) \in I_{op}$ for every $(x_1, x_2, \ldots, x_k) \in I_1 \times I_2 \times \cdots \times I_k$. By transitivity of set inclusion, if we implement a function g as a composition of such operations in interval arithmetic, then it is guaranteed that the range of g is included in the output interval I_g.

This property is useful when it comes to certifying the nonnegativity of g. Indeed, if the output interval I_g lies entirely in $[0, \infty)$, then we can establish that g is a nonnegative function on the given input intervals. However, since the computation is usually not exact, to maintain correctness, I_g will also contain elements that are not in the range of g. In particular, if g attains 0, then we cannot hope to certify the nonnegativity of g with interval arithmetic unless some very special conditions on g allow for exact evaluation.

Even in the case where $\inf(g) > 0$, I_g may still contain negative elements. For example, if $g = g_1 + g_2$, then I_g might be obtained by adding I_{g_1} and I_{g_2}. This will imply that $\sup(I_{g_1}) + \sup(I_{g_2}) \in I_g$, while in reality g_1 and g_2 may attain maximum/supremum on very different inputs. This issue can be resolved via a simple *divide-and-conquer* algorithm. Whenever the check on I_g is inconclusive, i.e., it contains both positive and negative numbers, then we split one of the input intervals into halves, and recursively apply the same computation to each half. This is like using a microscope: if we cannot see a region clearly, we zoom in to get a better view.

The pseudocode of the algorithm is presented in Algorithm 3. The CHECKVALIDITY function checks if there exists a valid configuration in $I_1 \times I_2 \times I_{1,2}$, i.e., a configuration that satisfies all triangle inequalities, and returns true if it does. If CHECKVALIDITY returns false, then the algorithm returns true, since in this case the region consists entirely of invalid configurations and there is nothing to check. Otherwise,

Algorithm 3 Interval arithmetic verification algorithm

```
1: procedure CheckRatio($I_1$, $I_2$, $I_{1,2}$)
2:     if CheckValidity($I_1, I_2, I_{1,2}$) = FALSE then
3:         return TRUE
4:     end if
5:     $I \leftarrow$ IntervalArithmeticEvaluate($I_1, I_2, I_{1,2}$).
6:     if $I \subseteq [0, \infty)$ then
7:         return TRUE
8:     else if $I \subseteq (\infty, 0)$ then
9:         return FALSE
10:    else
11:        if $|I_1| = \max(|I_1|, |I_2|, |I_{1,2}|)$ then
12:            Split $I_1$ into two equal-length sub-intervals $I_1 = I_1^l \cup I_1^r$.
13:            return CheckRatio($I_1^l, I_2, I_{1,2}$) $\wedge$ CheckRatio($I_1^r, I_2, I_{1,2}$)
14:        else if $|I_2| = \max(|I_1|, |I_2|, |I_{1,2}|)$ then
15:            Split $I_2$ into two equal-length sub-intervals $I_2 = I_2^l \cup I_2^r$.
16:            return CheckRatio($I_1, I_2^l, I_{1,2}$) $\wedge$ CheckRatio($I_1, I_2^r, I_{1,2}$)
17:        else
18:            Split $I_{1,2}$ into two equal-length sub-intervals $I_{1,2} = I_{1,2}^l \cup I_{1,2}^r$.
19:            return CheckRatio($I_1, I_2, I_{1,2}^l$) $\wedge$ CheckRatio($I_1, I_2, I_{1,2}^r$)
20:        end if
21:    end if
22: end procedure
```

the algorithm continues to compute an interval I, using the IntervalArithmeticEvaluate subroutine, such that

$$\forall \theta \in I_1 \times I_2 \times I_{1,2}, \qquad \mathop{\mathbb{E}}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha \cdot \mathsf{Comp}(\theta) \in I.$$

The algorithm then checks if I is entirely non-negative or entirely negative, in which cases we can decide that either the ratio is achieved over the entire region, or there exists a valid configuration that violates the ratio, and exit the algorithm accordingly. Otherwise, I consists of both positive and negative values, but the negative values may come from evaluation of invalid configurations, or more intrinsically the error produced by interval arithmetic itself. In this case, we subdivide the longest interval into two equal-length sub-intervals and recursively apply the algorithm, as explained earlier.

We implemented this verification algorithm in C using the interval arithmetic library Arb [40]. Specific advantages of this library is that it has rigorous implementations of the error function [41] as well as a routine for rigorous numerical integration [42]. To speed up the computation, we split the various tasks between cores using GNU Parallel [43]. We obtain the following lemma.

Lemma IV.5. *$\tilde{F}$ achieves an approximation ratio of* 0.87447 *on all* MAX DI-CUT *configurations with completeness at least* 10^{-6}.

We address the requirement on completeness in the next subsection.

2) *Removing the completeness requirement and a proof of Theorem I.3*

As we discussed, interval arithmetic in general cannot certify nonnegativity of a function which attains 0. Unfortunately, the function that we care about, $\mathbb{E}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha \cdot \mathsf{Comp}(\theta)$, does attain 0, regardless of the choice of r, as the following proposition shows.

Proposition IV.6. *Let $\theta = (b_i, b_j, b_{ij})$ be a configuration with $b_i = b_j = b$ and $\rho(\theta) = 1$. Then for any f,*

$$\mathsf{Sound}(\theta, f) \;=\; \mathsf{Comp}(\theta) \;=\; 0\,.$$

Proof. Since $\rho(\theta) = 1$, we have

$$b_{ij} \;=\; b_i b_j + \rho\sqrt{1-b_i^2}\sqrt{1-b_j^2} \;=\; b^2 + (1-b^2) \;=\; 1$$

and

$$\mathsf{Comp}(\theta) \;=\; \frac{1 + b_i - b_j - b_{ij}}{4} \;=\; \frac{1 + b - b - 1}{4} \;=\; 0\,.$$

For soundness, we have $\mathsf{Sound}(\theta, f) = \Phi_{-\rho}(f(b_i), -f(b_j)) = \Phi_{-\rho}(f(b), -f(b))$. Since $\rho = 1$, this is equal to $\Pr_{X\sim N(0,1)}[X \le f(b) \wedge -X \le -f(b)] = \Pr_{X\sim N(0,1)}[X = f(b)] = 0$. □

Luckily, on configurations with small completeness, it is known that independent rounding, which assigns true to each variable independently with probability 1/2, does very well. Indeed, this rounding scheme satisfies each MAX DI-CUT constraint with probability 1/4 on every configuration. This implies that $\tilde{F}$ combined with the independent rounding will achieve a good approximation ratio over all DI-CUT configurations.

Proof of Theorem I.3. Consider the rounding algorithm where we use the $\mathcal{THRESH}$ rounding scheme $\tilde{F}$ with probability $(1 - 10^{-5})$ and independent rounding with probability 10^{-5}. We show that this algorithm achieves a ratio of 0.87446 on all configurations of MAX DI-CUT.

Let θ be a DI-CUT configuration. If $\mathsf{Comp}(\theta) \ge 10^{-6}$, then by Theorem IV.5, we achieve a ratio of at least $0.87447 \times (1 - 10^{-5}) > 0.87446$. If $\mathsf{Comp}(\theta) < 10^{-6}$, then independent rounding contributes a soundness of $0.25 \times 10^{-5} = 2.5 \times 10^{-6} > 0.87446 \cdot \mathsf{Comp}(\theta)$. □

3) *Further optimizations*

To further speed up the computation, we compute partial derivatives of $\mathbb{E}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha \cdot \mathsf{Comp}(\theta)$, and reduce an interval to its boundary point if the corresponding partial derivative is nonnegative or nonpositive.

For example, if we have for every $\theta \in I_1 \times I_2 \times I_{1,2}$

$$\frac{\partial}{\partial b_1}\left(\mathop{\mathbb{E}}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha \cdot \mathsf{Comp}(\theta)\right) \geq 0 \ ,$$

and $I_1 = [l, r]$, then to certify the nonnegativity of $\mathbb{E}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)] - \alpha \cdot \mathsf{Comp}(\theta)$, it is sufficient to check

$$\forall\theta \in \{l\}\times I_2\times I_{1,2}, \quad \mathop{\mathbb{E}}_{f\sim\tilde{F}}[\mathsf{Sound}(\theta, f)]-\alpha\cdot\mathsf{Comp}(\theta) \ \geq \ 0 \ .$$

We remark that we only perform this optimization in regions that are entirely valid, i.e., consisting only of valid configurations. This is because otherwise we may reduce the region to an invalid subregion, on which the program returns true without checking the ratio.

4) Implementation details

To compute the soundness, we need to evaluate bivariate Gaussian distributions of the form $\Phi_\rho(t_1, t_2)$. However, Arb only has implementation of one-dimensional integration. To overcome this, we use the following formula from Drezner and Wesolowsky [29], which transforms $\Phi_\rho(t_1, t_2)$ into a one-dimensional integral:

$$\Phi_\rho(t_1, t_2) \ = \ \frac{1}{2\pi}\int_0^\rho \frac{1}{\sqrt{1-r^2}}\exp\left(-\frac{t_1^2 - 2rt_1t_2 + t_2^2}{2(1-r^2)}\right) dr + \Phi(t_1)\Phi(t_2) \ .$$

Another potential issue is numerical stability. Computing ρ from (b_i, b_j, b_{ij}) involves division by $\sqrt{(1-b_i^2)(1-b_j^2)}$, which can be unstable when b_i or b_j is close to ± 1. In the actual implementation, we overcome this by representing a configuration using (b_i, b_j, ρ).

V. A NEW APPROXIMATION ALGORITHM FOR MAX 2-AND

Recall that $\mathcal{THRESH}$ rounding schemes for MAX 2-AND are nearly identical to those for MAX DI-CUT, except that the rounding schemes for MAX 2-AND are required to be *odd* functions. It is easy to enforce in the discovery algorithm that the family of piecewise-linear functions we consider are odd (in fact, the oracle runs quicker as the number of free parameters is cut in half). Empirically, we found a "raw" distribution of 15 rounding functions which attains a ratio of approximately 0.8741. Using a clean-up procedure similar to that for MAX DI-CUT, we were able to simplify it to another distribution $\tilde{F}'$ with only 3 functions. See Table II for details.

Using the same interval arithmetic algorithm used for MAX DI-CUT, we obtain the following result.

Lemma V.1. *$\tilde{F}'$ achieves an approximation ratio of* 0.87415 *on all* MAX 2-AND *configurations with completeness at least* 10^{-6}.

We can then use the same proof idea as that in Section IV-D2 to get rid of the completeness requirement and obtain the lower bound of 0.87414 for MAX 2-AND, as claimed in Theorem I.4.

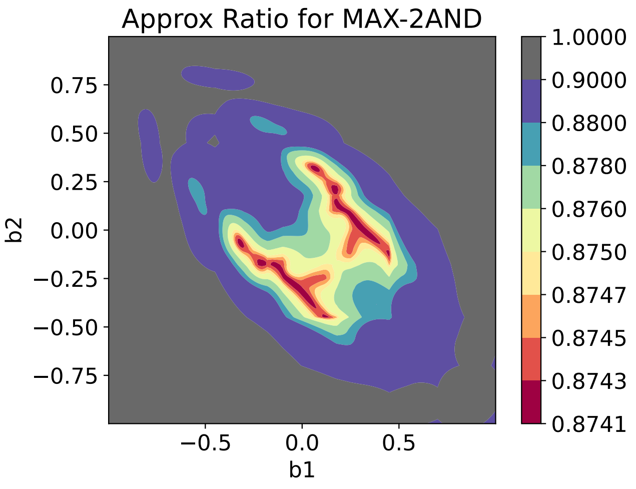


Fig. 5. This plot is a contour plot of the performance of the $\mathcal{THRESH}$ scheme for MAX 2-AND with 3 piecewise-linear rounding functions for various choices of b_1 and b_2 (with an approximately worst-case choice of b_{12}) selected.

VI. CONCLUSION

We used a "computational lens" to obtain a much better, and an almost complete, understanding of the MAX DI-CUT and MAX 2-AND problems. Insights gained from numerical experiments yielded a completely analytical new upper bound for MAX DI-CUT that can be verified by hand (see Section III), as well as new lower bounds, i.e., new approximation algorithms, for MAX DI-CUT and MAX 2-AND, for which we obtain a rigorous computer-assisted analysis (see Section IV and Section V).

We have established that the MAX DI-CUT problem has its own approximation ratio by strictly separating it from MAX 2-AND and MAX CUT (assuming the unique games conjecture). Fundamental to our approach was the use of algorithmic discovery to identify both difficult instances of MAX DI-CUT and MAX 2-AND as well as discovering $\mathcal{THRESH}$ rounding schemes which improve on the 20+ year state of the art.

As discussed in Section IV, assuming the unique games conjecture and Austrin's positivity conjecture, the optimal $\mathcal{THRESH}$ schemes[8] achieve $\alpha_{\text{DI-CUT}}$ and α_{2AND} for MAX DI-CUT and MAX 2-AND, respectively. We demonstrated a computational procedure which helps us to approximate $\alpha_{\text{DI-CUT}}$ and α_{2AND} to greater precision than previously known. However, a proper theoretical understanding is still missing. In particular:

Theoretical understanding of the optimal $\mathcal{THRESH}$ scheme. Currently, we lack a satisfactory explanation of why the secondary functions in the currently best-known MAX DI-CUT and MAX 2-AND $\mathcal{THRESH}$ schemes take on the shapes they do. Perhaps one can prove that the optimal functions must satisfy particular constraints (such as in the calculus

[8]Or more precisely a limiting sequence of finite, bounded $\mathcal{THRESH}$ schemes.

	f_1	f_2	f_3
prob	0.998105	0.001126	0.000769
-1.000000	-1.585394	0.934459	0.163540
-0.700000	-0.870350	0.443616	-0.212976
-0.450000	-0.512239	0.675617	-1.435794
-0.300000	-0.332896	-1.446206	0.289432
-0.250000	-0.274526	-1.495506	2.000000
-0.179515	-0.193131	-0.382870	-0.492446
-0.164720	-0.176869	0.015196	-0.933550
-0.100000	-0.107901	2.000000	-1.568231
0.000000	0.000000	0.000000	0.000000
0.100000	0.107901	-2.000000	1.568231
0.164720	0.176869	-0.015196	0.933550
0.179515	0.193131	0.382870	0.492446
0.250000	0.274526	1.495506	-2.000000
0.300000	0.332896	1.446206	-0.289432
0.450000	0.512239	-0.675617	1.435794
0.700000	0.870350	-0.443616	0.212976
1.000000	1.585394	-0.934459	-0.163540

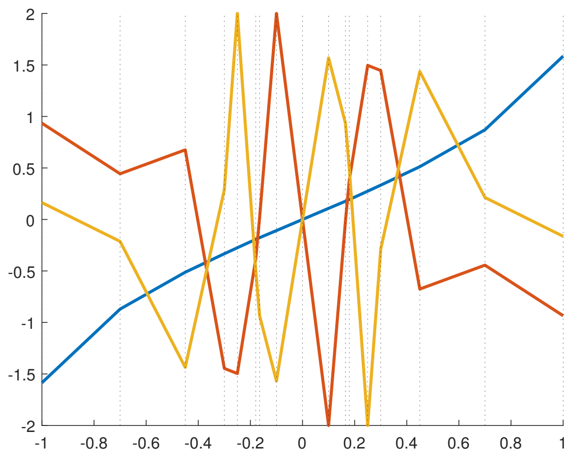


TABLE II
A $\mathcal{THRESH}$ ROUNDING SCHEME THAT GIVES A RIGOROUSLY VERIFIED APPROXIMATION RATIO OF AT LEAST 0.87414 FOR MAX 2-AND. (THE ACTUAL RATIO IS PROBABLY ABOUT 0.874202.) THE SCHEME USES THREE PIECEWISE-LINEAR ODD ROUNDING FUNCTIONS f_1, f_2, f_3 DEFINED ON 17 CONTROL POINTS. A PLOT OF THE FUNCTIONS IS GIVEN ON THE RIGHT.

of variations), or at least provide a satisfactory understand of the second-order affect these functions have.

Theoretical understanding of the hardest configurations. Likewise, we do not understand the structure of the hardest distributions of configurations for MAX DI-CUT and MAX 2-AND. In the full version, we demonstrate that some rather complex distributions appear to give increasingly better upper bounds for $\alpha_{\text{DI-CUT}}$ and α_{2AND}. Would it be possible to theoretically describe what the hardest configurations are? It is not clear whether the hardest distribution should even have finite support. Properly describing the hardest distributions would also resolve Austrin's positivity conjecture.

REFERENCES

[1] M. X. Goemans and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *Journal of the ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.

[2] U. Feige and M. Goemans, "Approximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT," in *Proceedings Third Israel Symposium on the Theory of Computing and Systems*. IEEE, 1995, pp. 182–189.

[3] S. Matuura and T. Matsui, "New approximation algorithms for MAX 2SAT and MAX DICUT," *Journal of the Operations Research Society of Japan*, vol. 46, no. 2, pp. 178–188, 2003.

[4] M. Lewin, D. Livnat, and U. Zwick, "Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems," in *International Conference on Integer Programming and Combinatorial Optimization*. Springer, 2002, pp. 67–82.

[5] H. Karloff and U. Zwick, "A 7/8-approximation algorithm for MAX 3SAT?" in *Proc. of 38th FOCS*. IEEE, 1997, pp. 406–415.

[6] U. Zwick, "Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint." in *Proc. of 9th SODA*, 1998, pp. 201–210.

[7] G. Andersson and L. Engebretsen, "Better approximation algorithms for set splitting and not-all-equal SAT," *Information Processing Letters*, vol. 65, no. 6, pp. 305–311, 1998.

[8] U. Zwick, "Outward rotations: A tool for rounding solutions of semidefinite programming relaxations, with applications to MAX CUT and other problems," in *Proc. of 31th STOC*. ACM, 1999, pp. 679–687. [Online]. Available: https://doi.org/10.1145/301250.301431

[9] E. Halperin and U. Zwick, "Approximation algorithms for MAX 4-SAT and rounding procedures for semidefinite programs," *Journal of Algorithms*, vol. 40, no. 2, pp. 184–211, 2001.

[10] T. Asano and D. P. Williamson, "Improved approximation algorithms for MAX SAT," *Journal of Algorithms*, vol. 42, no. 1, pp. 173–202, 2002.

[11] J. Zhang, Y. Ye, and Q. Han, "Improved approximations for max set splitting and max NAE SAT," *Discrete Applied Mathematics*, vol. 142, no. 1-3, pp. 133–149, 2004.

[12] A. Avidor, I. Berkovitch, and U. Zwick, "Improved approximation algorithms for MAX NAE-SAT and MAX SAT," in *Approximation and Online Algorithms, Third International Workshop, WAOA 2005*, ser. Lecture Notes in Computer Science, vol. 3879. Springer, 2005, pp. 27–40. [Online]. Available: https://doi.org/10.1007/11671411_3

[13] J. Brakensiek, N. Huang, A. Potechin, and U. Zwick, "On the mysteries of MAX NAE-SAT," in *Proc. of 32nd SODA*. SIAM, 2021, pp. 484–503. [Online]. Available: https://doi.org/10.1137/1.9781611976465.30

[14] S. Abbasi-Zadeh, N. Bansal, G. Guruganesh, A. Nikolov, R. Schwartz, and M. Singh, "Sticky brownian rounding and its applications to constraint satisfaction problems," *ACM Trans. Algorithms*, vol. 18, no. 4, oct 2022. [Online]. Available: https://doi.org/10.1145/3459096

[15] R. Eldan and A. Naor, "Krivine diffusions attain the Goemans–Williamson approximation ratio," 2019.

[16] K. Makarychev and Y. Makarychev, "Approximation Algorithms for CSPs," in *The Constraint Satisfaction Problem: Complexity and Approximability*, ser. Dagstuhl Follow-Ups, A. Krokhin and S. Zivny, Eds. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, vol. 7, pp. 287–325. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2017/6968

[17] J. Håstad, "Some optimal inapproximability results," *Journal of the ACM*, vol. 48, no. 4, pp. 798–859, 2001.

[18] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, "Proof verification and the hardness of approximation problems," *Journal of the ACM*, vol. 45, no. 3, pp. 501–555, 1998.

[19] L. Trevisan, G. B. Sorkin, M. Sudan, and D. P. Williamson, "Gadgets, approximation, and linear programming," *SIAM Journal on Computing*, vol. 29, no. 6, pp. 2074–2097, 2000.

[20] S. Khot, "On the power of unique 2-prover 1-round games," in *FOCS 2002*, 2002, pp. 767–775.

[21] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell, "Optimal inapproximability results for MAX-CUT and other 2-variable CSPs?" *SIAM Journal on Computing*, vol. 37, no. 1, pp. 319–357, 2007.

[22] E. Mossel, R. O'Donnell, and K. Oleszkiewicz, "Noise stability of functions with low influences: Invariance and optimality," *Annals of Mathematics*, pp. 295–341, 2010.

[23] P. Austrin, "Balanced MAX 2-SAT might not be the hardest," in *Proc. of 39th STOC*, 2007, pp. 189–197.

[24] ——, "Towards sharp inapproximability for any 2-CSP," *SIAM Journal on Computing*, vol. 39, no. 6, pp. 2430–2463, 2010.

[25] P. Raghavendra, "Optimal algorithms and inapproximability results for every CSP?" in *Proc. of 40th STOC*, 2008, pp. 245–254.

[26] ——, "Approximating NP-hard problems - efficient algorithms and their limits," Ph.D. dissertation, University of Washington, 2009.

[27] P. Raghavendra and D. Steurer, "How to round any CSP," in *Proc. of 50th FOCS*. IEEE, 2009, pp. 586–594.

[28] U. Zwick, "Computer assisted proof of optimal approximability results." in *Proc. of 13th SODA*, 2002, pp. 496–505.

[29] Z. Drezner and G. O. Wesolowsky, "On the computation of the bivariate normal integral," *Journal of Statistical Computation and Simulation*, vol. 35, no. 1-2, pp. 101–107, 1990.

[30] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[31] J. v. Neumann, "Zur theorie der gesellschaftsspiele," *Mathematische annalen*, vol. 100, no. 1, pp. 295–320, 1928.

[32] J. Nash, "Non-cooperative games," *Annals of mathematics*, pp. 286–295, 1951.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[34] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[35] A. Genz, "Numerical computation of multivariate normal probabilities," *J. Comp. Graph Stat.*, vol. 1, pp. 141–149, 1992.

[36] ——, "Comparison of methods for the computation of multivariate normal probabilities," *Computing Science and Statistics*, vol. 25, pp. 400–405, 1993.

[37] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[38] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.

[39] A. Domahidi, E. Chu, and S. Boyd, "Ecos: An socp solver for embedded systems," in *2013 European Control Conference (ECC)*. IEEE, 2013, pp. 3071–3076.

[40] F. Johansson, "Arb: efficient arbitrary-precision midpoint-radius interval arithmetic," *IEEE Transactions on Computers*, vol. 66, no. 8, pp. 1281–1292, 2017.

[41] ——, "Computing hypergeometric functions rigorously," *ACM Transactions on Mathematical Software (TOMS)*, vol. 45, no. 3, pp. 1–26, 2019.

[42] ——, "Numerical integration in arbitrary-precision ball arithmetic," in *International Congress on Mathematical Software*. Springer, 2018, pp. 255–263.

[43] O. Tange, "Gnu parallel - the command-line power tool," *;login: The USENIX Magazine*, vol. 36, no. 1, pp. 42–47, Feb 2011. [Online]. Available: https://www.gnu.org/s/parallel