# An Improved Approximation Algorithm for the Max-3-Section Problem

**Dor Katzelnick** #
The Henry and Marilyn Taub Faculty of Computer Science, Technion, Haifa, Israel

**Aditya Pillai** #
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA

**Roy Schwartz** #
The Henry and Marilyn Taub Faculty of Computer Science, Technion, Haifa, Israel

**Mohit Singh** #
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA

## Abstract

We consider the Max-3-Section problem, where we are given an undirected graph $G = (V, E)$ equipped with non-negative edge weights $w : E \to \mathbb{R}_+$ and the goal is to find a partition of $V$ into three equisized parts while maximizing the total weight of edges crossing between different parts. Max-3-Section is closely related to other well-studied graph partitioning problems, e.g., Max-Cut, Max-3-Cut, and Max-Bisection. We present a polynomial time algorithm achieving an approximation of 0.795, that improves upon the previous best known approximation of 0.673. The requirement of multiple parts that have equal sizes renders Max-3-Section much harder to cope with compared to, e.g., Max-Bisection. We show a new algorithm that combines the existing approach of Lassere hierarchy along with a random cut strategy that sufices to give our result.

**2012 ACM Subject Classification** Theory of computation → Approximation algorithms analysis; Theory of computation → Semidefinite programming

**Keywords and phrases** Approximation Algorithms, Semidefinite Programming, Max-Cut, Max-Bisection

**Digital Object Identifier** 10.4230/LIPIcs.ESA.2023.69

**Related Version** *Full Version*: https://arxiv.org/abs/2308.03516

**Funding** Mohit Singh and Aditya Pillai were supported in part by NSF CCF-2106444 and NSF CCF-1910423. Dor Katzelnick and Roy Schwartz receive funding from the European Union's Horizon 2020 research and innovation program under grant agreement no. 852870-ERC-SUBMODULAR.

**Acknowledgements** The authors would like to thank Uri Zwick for insightful discussions.

## 1 Introduction

In this paper we study the Max-3-Section problem: given an undirected graph $G = (V, E)$ equipped with non-negative edge weights $w : E \to \mathbb{R}_+$ the goal is to partition the vertex set $V$ into three equisized parts while maximizing the total weight of edges that cross between different parts. Max-3-Section is closely related to other classic graph partitioning problems, where given the same input as in Max-3-Section the goal is to output a partition of the vertex set $V$ (possibly given some problem specific constraint) while maximizing the total weight of edges that cross between different parts. Perhaps the most famous of these problems is Max-Cut, whose constraint is that the partition contains only two parts with no restriction on the size of these parts. Max-Cut is one of Karp's 21 NP-hard problems [11]


© Dor Katzelnick, Aditya Pillai, Roy Schwartz, and Mohit Singh;
licensed under Creative Commons License CC-BY 4.0

and in their seminal work Goemans and Williamson [9] presented an approximation of 0.8786 using semi-definite programming and random hyperplane rounding. It is known that the latter result is tight, assuming the unique games conjecture, as proved by Khot, Kindler, Mossel, and O'Donnell [12].

A natural problem that generalizes Max-Cut is Max-$k$-Cut, whose constraint is that the partition contains $k$ parts with no restriction on the size of these parts. A simple algorithm that returns a uniform random solution, i.e., every vertex is assigned independently and uniformly to one of the $k$ parts, achieves an approximation of $1-1/k$. Several works aimed at improving the guarantee of this simple algorithm, e.g., [6, 8, 4, 14]. For example, a notable case that attracted much attention is the Max-3-Cut problem [6, 8, 4, 14], whose best approximation is 0.836 and was given by Goemans and Williamson [8]. Interestingly, the latter guarantee is worse than the approximation known for Max-Cut. For general values of $k$, it was shown by Frieze and Jerrum [6] that an approximation of $1-1/k+\Theta(\ln k/k^2)$ can be achieved. Further improvements for the approximation guarantee were presented by de Klerk, Pasechnik and Warners [4].

Adding a single global constraint to Max-Cut that requires both parts to be of equal size leads to the classic problem of Max-Bisection. Max-Bisection was extensively studied throughout the years, e.g., [6, 16, 10, 5, 15, 2]. This sequence of works currently culminates with the works of Raghavendra and Tan [15], who present an approximation of 0.85 that is based on rounding a Lasserre hierarchy semi-definite program, which was subsequently improved to 0.877 by Austrin, Benabbas, and Georgiou [2] who improved the former rounding. The question whether one can obtain for Max-Bisection the same approximation guarantee that is known for Max-Cut remains a tantalizing open problem.

Both Max-3-Section and Max-Bisection are captured by the Max-$k$-Section problem [1, 7, 13, 3], which falls in the above broad family of graph partitioning problems and whose constraint is that the partition contains $k$ equisized parts. Similarly to Max-$k$-Cut, it is known that the simple algorithm that returns a uniform random solution achieves an approximation of $1-1/k$. Thus, it is no surprise that the goal of past research, e.g., [1, 7, 13], was to improve upon this guarantee. For the special case of Max-3-Section, Ling [13] presented an approximation of 0.6733 that is based on rounding a semi-definite programming relaxation similarly to Max-3-Cut [8]. For general values of $k$, Andersson [1] presented an approximation of $1-1/k+\Theta(1/k^3)$, again by rounding a suitable semi-definite program relaxation. Additionally, Gaur, Krishnamurti, and Kohli [7] presented a local search algorithm for a more general problem in which each part has a (possibly different) limit on its size.

The main focus of this work is the Max-3-Section problem. For $k=2$, the best known approximation for Max-Bisection (which is essentially Max-Cut with a single global constraint on the size of the first part in the partition) equals 0.877 and is (almost) identical to the best possible approximation of 0.878 for Max-Cut. However, when $k=3$, the best known approximation for Max-3-Section (which is essentially Max-3-Cut with two global constraints on the size of the first two pieces in the partition) equals 0.6733 and is far from the best known approximation of 0.836 for Max-3-Cut. Moreover, the former approximation guarantee of 0.6733 for Max-3-Section only slightly improves upon the $2/3$ approximation guarantee of the trivial algorithm that simply returns a uniform random solution. Thus, we aim to understand and minimize the gap that exists for $k=3$.

## 1.1 Our Results and Techniques

We present the following main algorithmic result for the Max-3-Section problem:

**Theorem 1.** *There is a polynomial time algorithm for Max-3-Section that achieves an approximation guarantee of at least* 0.795.

It is important to note that the approximation guarantee of the above theorem improves upon the previous best known algorithm for Max-3-Section [13] that achieves an approximation of 0.6773. As proving the exact approximation guarantee of our algorithm is an involved task (for reasons that will be clear soon), we also present the following conjecture that is based on numerical evidence:

**Conjecture 2.** *The algorithm presented to prove Theorem 1 achieves an approximation of* 0.8192 *for Max-3-Section.*

We further show how the algorithm we present for proving Theorem 1 can be extended to general values of $k$. First, we prove that the algorithm, alongside its analysis, cannot provide an improved approximation as $k$ increases (when compared to its approximation for the case of $k = 3$). Second, using numeric evidence we conjecture that the approximation of the algorithm remains 0.8192 for $k$ ranging from 3 up to 6. Hence, the above gives rise to the following extended conjecture:

**Conjecture 3.** *There is a polynomial time algorithm achieving an approximation of* 0.8192 *for Max-k-Section for* $k = 3, 4, 5$.

The above conjecture provides, for $k = 4, 5$, an improved approximation when compared to the previous best known result. The current best is a small improvement over the trivial randomized approximation algorithm by Andersson [1] which achieves an approximation of $1 - 1/k + \Theta(k^{-3})$.

When considering our approach to Max-3-Section we focus on two of its closely related problems: Max-3-Cut and Max-Bisection and examine the approaches used to design and analyze algorithms for both. Let us start with Max-3-Cut. The approach used to obtain the current best known algorithms for Max-3-Cut, e.g., [8, 4, 14], is based on the random hyperplane rounding method due to Goemans and Williamson [9] (as well as extensions of this method) of a semi-definite programming relaxation. This approach, when applied to Max-3-Section, suffers a significant drawback since it does not preserve marginal values (a marginal value is the likelihood the relaxation assigns to the event that vertex $u$ belongs to part $i$). Specifically, the expected number of vertices assigned to every part by the rounding algorithm might be incomparable to $n/3$ and therefore applying these ideas as was done by Ling [13] leads only to a minor improvement over the random solution algorithm. It is worth noting that this drawback is already present when considering Max-Bisection.

Let us now consider Max-Bisection, and specifically we focus on the approach of Raghavendra and Tan [15] (as well as Austrin, Benabbas, and Georgiou [2] who build upon [15]). First, a Lasserre hierarchy of a natural semi-definite programming relaxation for Max-Bisection is solved. Second, a rounding procedure that preserves the marginal values is applied to obtain a subset $C_1 \subseteq V$ of vertices such that: (1) there are suficiently many edges crossing the cut $C_1$ defines; and (2) $C_1$ contains (roughly) $n/2$ vertices. Third, the solution is re-balanced to obtain a perfect bisection, i.e., ensuring that $|C_1| = n/2$ without a significant loss in the number of edges crossing between $C_1$ and $C_2 = V \setminus C_1$.

There are two main dificulties when considering this approach in the context of Max-3-Section. The first dificulty stems from the fact that we have three parts in Max-3-Section, whereas in Max-Bisection there are only two parts. Therefore, if we use the above rounding procedure to find $C_1 \subseteq V$ of size (roughly) $n/3$ with suficiently many edges crossing the cut $C_1$ defines, it is not clear how to recurse and further partition $V \setminus C_1$. We note that in Max-Bisection no recursion is needed since $C_2$ is chosen to be $V \setminus C_1$. However, in Max-3-Section $V \setminus C_1$ still needs to be partitioned into $C_2$ and $C_3$. Our solution to this

dificulty is to *condition* the marginal values of vertices remaining in $V \setminus C_1$ on the fact that each remaining vertex was not chosen to $C_1$. Such a conditioning, intuitively, ensures that we preserve marginal values overall while recursing on $V \setminus C_1$.

The second dificulty in applying this apporach arises from the following observation: we can show that if one first creates part $C_1$, and then creates part $C_2$ (and part $C_3$ is all remaining vertices), then if the analysis is performed edge-by-edge (as is the case in both [15, 2]) no approximation better than 0.7192 can be achieved. Specifically, we present a configuration of the vectors that correspond to the endpoints of an edge that satisfy: (1) the vectors are feasible for the semi-definite programming relaxation; and (2) the ratio between the probability of this edge being separated by the rounding algorithm and the contribution of its vectors to the objective function of the relaxation is at most 0.7192 (refer to Section 4 for a formal definition of a configuration and to Observation 35 in the full version of the paper). An approximation of 0.7192, if possible given the above approach, improves the current best known approximation of 0.6733 [13]. However, we aim for a much larger improvement. Our solution to this dificulty is to *uniformly permute* the order in which the parts are generated. Since the approach based on [15, 2] preserves marginals, this permutation allows us to better cope with problematic configurations. It is important to note that a permutation is meaningless for Max-Bisection, since whether a vertex belongs to $C_1$ immediately implies whether it belongs to $C_2$ and vice versa. In Max-3-Section this is obviously not the case.

Thus, following the above discussion, our approach builds upon the approach for Max-Bisection with two added ingredients. The first is altering the marginal values the semi-definite programming relaxation provides via appropriate conditioning when recursing. The second is uniformly permuting the order in which the rounding algorithm generates the parts. We note that these two added ingredients introduce two main additional obstacles. The first obstacle relates to the last re-balancing step. In both [15, 2] the re-balancing succeeds since it is proved that with a high probability each part by itself is close to being the desired size. The method this is proved is by bounding the variance of the size of each part alone. However, in our approach for Max-3-Section the bound on the variance of the size of a given part depends on the other parts as well. This introduces technical issues and hence bounding of the variance requires much care. The second obstacle relates to the computer assisted proof via *branch and bound* method we employ in order to lower bound the performance ratio of our algorithm. The expression of the separation probability of an edge by the rounding algorithm is involved, as both marginal values are altered when recursing and we employ a random permutation over the order in which the parts are generated. Moreover, a configuration describing how the semi-definite programming relaxation encodes an edge involves 7 different vectors (see Sections 3 and 4). Thus, we had to incorporate many technical ingredients, e.g., analytically bounding the gradient of the separation probability and restricting the search to specific type of configurations while analytically bounding the error this incurs, to make the computer assisted proof terminate faster. This results in about 150,000 hours of CPU, which is roughly 20 CPU years, to prove Theorem 1.

## 1.2 Additional Related Work

The Max-Bisection problem has a long and rich history. Frieze and Jerrum [6] presented an approximation of 0.6514 based on rounding a semi-definite program. Later on, Ye [16], Halperin and Zwick [10], and Feige and Langberg [5] further improved the approximation guarantee to 0.699, 0.7016, and 0.7027, respectively. They achieved the above by strengthening the semi-definite programming relaxation, e.g., by adding triangle inequality constraints,

and presenting better rounding methods. The next leap in approximating Max-Bisection came with the work of Raghavendra and Tan [15]. They utilized a higher-level Lasserre hierarchy semi-definite program, together with an elegant rounding algorithm, and obtained a 0.85-approximation. Later on, Austrin, Benabbas, and Georgiou [2] showed an improved rounding algorithm, pushing the approximation guarantee up to 0.877. It should be noted that the latter is very close to the best possibloe approximation of 0.878 for Max-Cut.

Focusing on Max-$k$-Cut, Frieze and Jerrum [6] presented an approximation algorithm with better guarantee than the naive random algorithm. They utilized a semi-definite program relaxation alongside an elegant rounding algorithm that samples $k$ random vectors and assigns every vertex $v \in V$ to the cluster of the random vector that is closest to $v$'s vector in the relaxation. Goemans and Williamson [8] presented an improved approximation of 0.836 for Max-3-Cut by using a complex semi-definite program. In [4], de Klerk, Pasechnik, and Warners presented further improved bounds for Max-$k$-Cut. Please refer to Table 1 by Newman [14] for a summary of approximation guarantees.

### 1.3 Paper Organization

We start by presenting preliminary definitions in Section 2. Next, in Section 3 we present our semi-definite program for Max-3-Section and show that it can be strengthened to obtain a solution which is globally uncorrelated. In Section 4 we present our rounding algorithm and its analysis. To obtain a bound on the approximation guarantee of our rounding algorithm, we present an analysis which is based on a computer-assisted proof in Section 5.
Furthermore, we discuss the generalization of our algorithm, and its numerical estimation, in Section 6. Missing proofs appear in the full version of the paper.

## 2 Preliminaries

We denote by $\Phi : \mathbb{R} \to [0,1]$ the cumulative distribution function of the normal gaussian distribution and by $\Phi^{-1} : [0,1] \to \mathbb{R}$ its inverse. Specifically, if $R \sim N[0,1]$ then: (1) $\forall x \in \mathbb{R}$: $\Pr[R \leq x] = \Phi(x)$ (or equivalently $\Pr[R \geq x] = 1 - \Phi(x)$); and (2) $\forall x \in [0,1]$: $\Pr[R \leq \Phi^{-1}(x)] = x$ (or equivalently $\Pr[R \geq \Phi^{-1}(x)] = 1 - x$). Moreover, we say that a vector $\mathbf{g}$ is a *random gaussian vector* if its coordinates are i.i.d standard gaussian $N(0,1)$ random variables.

We denote by $\Gamma_t : [0,1]^2 \to [0,1]$ the probability that a standard bi-variate gaussian distribution with correlation $t$ has both its coordinates at most the given quantiles, i.e.,

$$\forall q_1, q_2 \in [0,1] : \Gamma_t(q_1, q_2) = \Pr[X \leq \Phi^{-1}(q_1), Y \leq \Phi^{-1}(q_2)], \quad \begin{matrix} X \\ Y \end{matrix} \sim N\left(\begin{matrix} 0 \\ 0 \end{matrix}, \begin{matrix} 1 & t \\ t & 1 \end{matrix}\right).$$

We define the mutual information between two random variables.

**Definition 4.** *Let $X, Y$ be jointly distributed random variables taking values in $[q]$. The mutual information of $X$ and $Y$ is defined as*

$$I(X,Y) = \sum_{i,j \in [q]} \Pr(X = i, Y = j) \log \frac{\Pr(X = i, Y = j)}{\Pr(X = i)\Pr(Y = j)}.$$

For any two disjoint subsets of vertices $A, B \subseteq V$, we denote by $\delta(A,B)$ the collection of edges having one endpoint in $A$ and another endpoint in $B$. Hence, $|\delta(A,B)|$ denotes the number of edges crossing between $A$ and $B$.[1]

[1] For simplicity of presentation, we assume from this point onward that the graph is unweighted. All of our results apply to graphs equipped with non-negative edge weights, in which case one should substitute $|\delta(A,B)|$ with the total weight of edges crossing between $A$ and $B$.

## 3 The SDP Relaxation for Max-3-Section

In this section, we present a semi-definite programming (SDP) formulation and prove that it is a relaxation for the Max-3-Section problem. Similarly to previous works on Max-Bisection, e.g., [2, 15], we strengthen this formulation and obtain additional properties that will be useful to our rounding algorithm. We define the following SDP formulation for Max-3-Section:

$$\text{(SDP)} \quad \text{maximize} \quad \sum_{e=(u,v)\in E} \left(1 - \sum_{i=1}^{3} \mathbf{y}_u^i \bullet \mathbf{y}_v^i\right) \tag{1}$$

$$\text{s.t.} \quad \|\mathbf{y}_\varnothing\|^2 = 1 \tag{2}$$

$$\|\mathbf{y}_v^i\|^2 = \mathbf{y}_\varnothing \bullet \mathbf{y}_v^i \qquad \forall v \in V, \forall i = 1,2,3 \tag{3}$$

$$\|\mathbf{y}_v^1\|^2 + \|\mathbf{y}_v^2\|^2 + \|\mathbf{y}_v^3\|^2 = 1 \qquad \forall v \in V \tag{4}$$

$$\mathbf{y}_v^i \bullet \mathbf{y}_v^j = 0 \qquad \forall v \in V, i = j \tag{5}$$

$$\mathbf{y}_u^i \bullet \mathbf{y}_v^j \geq 0 \qquad \forall u,v \in V, \forall i,j = 1,2,3 \tag{6}$$

$$\sum_{v\in V} \|\mathbf{y}_v^i\|^2 = n/3 \qquad \forall i = 1,2,3 \tag{7}$$

We note that in the above and what follows, for every vector $\mathbf{y}$ a square norm of a vector $\|\mathbf{y}\|^2$ is with respect to the $\ell_2$ (Euclidean) norm and equals $\mathbf{y} \bullet \mathbf{y}$. Next, we prove that the formulation is a relaxation for our problem. Intuitively, for every vertex $v \in V$ the formulation SDP assigns a distribution over the three clusters via the vectors $\mathbf{y}_v^1$, $\mathbf{y}_v^2$, and $\mathbf{y}_v^3$. Specifically, $\mathbf{y}_\varnothing$ is a unit vectors (Constraint (2)) that denotes *true* whereas the zero vector (that does not appear explicitly in SDP) denotes *false*. Each vector $\mathbf{y}_v^i$ indicates how much vertex $v$ is likely to be assigned to the $i^{\text{th}}$ cluster by SDP. Hence, $\|\mathbf{y}_v^i\|^2$, or equivalently $\mathbf{y}_v^i \bullet \mathbf{y}_\varnothing$ (see Constraint (3)), denotes the marginal probability of assigning vertex $v$ to the $i^{\text{th}}$ cluster by SDP. For every vertex $v \in V$, the sum of these marginal probabilities needs to sum up to one (Constraint 4). Since every vertex $v \in V$ can be assigned to a single cluster in any integral solution, SDP enforces that the vectors $\mathbf{y}_v^i$ and $\mathbf{y}_v^j$ for $i = j$ are orthogonal (Constraint (5)). Intuitively, the joint probability SDP assigns for vertices $u$ and $v$ to belong to the $i^{\text{th}}$ and $j^{\text{th}}$ clusters, respectively, is non-negative (Constraint (6)). Finally, since the three clusters are required to be of size $n/3$ each Constraint (7) is added to SDP. When focusing on the objective of SDP (see (1)), for every edge $(u, v) \in E$ the inner product $\mathbf{y}_u \bullet \mathbf{y}_v$ intuitively indicates the joint probability of both $u$ and $v$ to be assigned to the $i^{\text{th}}$ cluster by SDP. Therefore, intuitively $1 - \mathbf{y}_u^1 \bullet \mathbf{y}_v^1 - \mathbf{y}_u^2 \bullet \mathbf{y}_v^2 - \mathbf{y}_u^3 \bullet \mathbf{y}_v^3$ indicates the likelihood of separating an edge $(u, v)$ by SDP. Thus, this is the objective of SDP.

The following lemma proves that SDP is a relaxation to the Max-3-Section problem, i.e., the value of an optimal solution $\text{OPT}_{\text{SDP}}$ to SDP is an upper bound on the value of an integral optimal solution OPT.

**Lemma 5.** *Given an instance of Max-3-Section let* $\text{OPT}_{\text{SDP}}$ *be the value of an optimal solution of* SDP *(1) and* OPT *be the value of an optimal integral solution. Then,* $\text{OPT}_{\text{SDP}} \geq \text{OPT}$.

**Proof.** Let $\{C_1^*, C_2^*, C_3^*\}$ be an optimal solution for the given instance of Max-3-Section whose value is OPT. Construct the following vector solution to SDP. First, fix an arbitrary unit vector $\mathbf{y}_\varnothing$. Second, for every $v \in V$ define $\mathbf{y}^i{}_v$ to be the zero vector if $v \notin C_i^*$ and $\mathbf{y}^i{}_v = \mathbf{y}_\varnothing$ if $v \in C_i^*$. One may notice that all the constraints hold for this solution and the value of the objective of SDP equals the value of the optimal solution $\{C_1^*, C_2^*, C_3^*\}$. Hence, $\text{OPT}_{\text{SDP}} \geq \text{OPT}$. ∎

For simplicity of presentation, we denote by $Y$ a solution to SDP. Thus, $Y$ consists of $\{\mathbf{y}_u^i\}_{u\in V, i=1,2,3}$ and $\mathbf{y}_\emptyset$. A useful property that any feasible solution $Y$ to SDP satisfies is that $\mathbf{y}_u^1 + \mathbf{y}_u^2 + \mathbf{y}_u^3$ always equals the vector $\mathbf{y}_\emptyset$. This is summarized in the following lemma.

**Lemma 6.** *Let $Y$ be a feasible solution to* SDP. *Then, for every vertex $u \in V$: $\mathbf{y}_u^1 + \mathbf{y}_u^2 + \mathbf{y}_u^3 = \mathbf{y}_\emptyset$.*

An immediate corollary of the above lemma is that for every pair of vertices $u, v \in V$: $\mathbf{y}_u^i \bullet (\mathbf{y}_v^1 + \mathbf{y}_v^2 + \mathbf{y}_v^3) = \|\mathbf{y}_u^i\|^2$ (via Constraint (3)).

## 3.1 Globally Uncorrelated Solution

Next we define when a solution $Y$ to SDP is globally uncorrelated following the framework of [15]. Global uncorrelation implies that our rounding algorithm returns a solution that has close to $\frac{n}{3}$ vertices in each part with high probability (see Lemma 15). The sizes then can be corrected with a minor loss in approximation ratio by randomly shifting the imbalanced vertices (see Lemma 16).

A simple fact about any SDP solution is the following: given any two vertices $u, v$, there exists a *local* probability distribution $\mu_{u,v}$ on $\{1,2,3\}^2$ such that $\Pr_{\mu_{u,v}}[X_u = i, X_v = j] = \langle \mathbf{y}_u^i, \mathbf{y}_v^j \rangle$ for every $i, j \in \{1,2,3\}$ and $\Pr_{\mu_{u,v}}[X_u = i] = \|\mathbf{y}_u^i\|^2$ for every $i \in \{1,2,3\}$. The distribution implies that, at least, locally the semi-definite program is a distribution over integral solutions that satisfy correct correlations. The last property states that the distributions are consistent on their intersection which can be at most one vertex.

**Definition 7.** *A solution $Y$ to* SDP *is $\varepsilon$-independent if $\mathrm{E}_{u,v}[I_{\mu_{u,v}}(X_u, X_v)] \leq \varepsilon$ where $\mu_{u,v}$ is the local probability distribution associated with vertices $u$ and $v$.*

The following lemma is an application of Theorem 4.6 from [15] to our SDP for Max-3-Section and it shows how to obtain a $\varepsilon$-independent solution. The algorithm proving Lemma 8 follows from solving the $\Theta(t^2)$-level Lasserre hierarchy semi-definite program for SDP and then inductively conditioning on variables.

**Lemma 8.** *There is an algorithm which, given an integer $t > 0$ and an instance of Max-3-Section, runs in time $n^{O(poly(t))}$ and outputs a set of vectors $Y$ consisting of $\{\mathbf{y}^i\}_{v\in V, i=1,2,3}$ and $\mathbf{y}_\emptyset$ such that:*

1. *$Y$ is a feasible solution to* SDP.
2. *The objective value of* SDP *(1) when plugging in $Y$ is at least $\mathrm{OPT}_{\mathrm{SDP}} - \frac{1}{t}$.*
3. *$Y$ is $\frac{1}{t}$-independent.*

# 4 The Rounding Algorithm

In this section we present our rounding algorithm for SDP, which appears in Algorithm 1. The algorithm receives as input a solution to SDP and outputs a partition $\{C_1, C_2, C_3\}$ of $V$ that: (1) has high value compared to the SDP value of the input solution; and (2) is (nearly) balanced, i.e., for every $i = 1,2,3$: $|C_i|$ is close to $n/3$. We will conclude the analysis by proving that one can re-balance the partition without a significant loss in the value of the solution.

In order to state the algorithm, we require the following definition. For every vertex $u \in V$ and $i = 1, 2, 3$ we denote by $\mathbf{z}_u^i$ the normalized component of $\mathbf{y}_u^i$ that is orthogonal to $\mathbf{y}_\emptyset$, i.e., $\mathbf{y}_u^i = \|\mathbf{y}_u^i\|^2 \mathbf{y}_\emptyset + \sqrt{\|\mathbf{y}_i^i\|^2 - \|\mathbf{y}_u^i\|^4}\,\mathbf{z}_u^i$. Equivalently,

$$\mathbf{z}_u^i = \frac{\mathbf{y}_u^i - \|\mathbf{y}_u^i\|^2 \mathbf{y}_\emptyset}{\sqrt{\|\mathbf{y}_u^i\|^2 - \|\mathbf{y}_u^i\|^4}}. \tag{8}$$

Clearly, $\mathbf{z}_u^i$ is a unit vector that is orthogonal to $\mathbf{y}_\emptyset$. We note that if the marginal of vertex $u$ and cluster $i$ is integral, i.e., $\|\mathbf{y}_u^i\|^2 \in \{0, 1\}$, then $\mathbf{z}_u^i$ is not defined. In this case one can simply choose an arbitrary unit vector in the space orthogonal to $\mathbf{y}_\emptyset$ to be $\mathbf{z}_u^i$.

**Algorithm 1** Max-3-Section Rounding Algorithm.

**Input:** solution $\{\mathbf{y}_u^i\}_{u \in V, i=1,2,3}$ and $\mathbf{y}_\emptyset$ to SDP.
**Output:** a partition of $V$ into three parts.

1 Draw uniformly at random a permutation $\pi \in \Sigma_3$.
2 Draw independently two random Gaussian vectors $\mathbf{g}_1$ and $\mathbf{g}_2$.
3 Define the following sets:

$$S_{\pi(1)} = \left\{ u \in V : \mathbf{z}_u^{\pi(1)} \bullet \mathbf{g}_1 \geq \Phi^{-1}\left(1 - \|\mathbf{y}_u^{\pi(1)}\|^2\right) \right\},$$
$$S_{\pi(2)} = \left\{ u \in V : \mathbf{z}_u^{\pi(2)} \bullet \mathbf{g}_2 \geq \Phi^{-1}\left(1 - \|\mathbf{y}_u^{\pi(2)}\|^2 / (1 - \|\mathbf{y}_u^{\pi(1)}\|^2)\right) \right\}.$$

4 Return $\{C_1, C_2, C_3\}$ where: $C_{\pi(1)} = S_{\pi(1)}$, $C_{\pi(2)} = S_{\pi(2)} \setminus S_{\pi(1)}$, $C_{\pi(3)} = V \setminus (S_{\pi(1)} \cup S_{\pi(2)})$.

We first prove that Algorithm 1 preserves the marginal probabilities of SDP, i.e., $\|\mathbf{y}_u^i\|^2$ is the probability vertex $u$ is assigned to cluster $C_i$. This is summarized in the following lemma.

**Lemma 9.** *For every $u \in V$ and $i = 1, 2, 3$, it holds that* $\Pr[u \in C_i] = \|\mathbf{y}_u^i\|^2$.

**Proof.** Fix a permutation $\pi \in \Sigma_3$, and let us calculate the probability that $u \in C_i$ conditioned on the event that $\pi$ was chosen in the first step of Algorithm 1. Hence, the following holds for $C_{\pi(1)}$:

$$\Pr_u[u \in C_{\pi(1)} | \pi] = \Pr_u[u \in S_{\pi(1)} | \pi] = \Pr_u\left[\mathbf{z}^{\pi(1)} \bullet \mathbf{g}_1 \geq \Phi^{-1}\left(1 - \|\mathbf{y}^{\pi(1)}\|^2\right) | \pi\right] = \|\mathbf{y}^{\pi(1)}\|^2.$$

We observe that the sets $S_{\pi(1)}$ and $S_{\pi(2)}$ are constructed with independent vectors $\mathbf{g}_1$ and $\mathbf{g}_2$. Therefore, similarly to the above, the following holds for $C_{\pi(2)}$:

$$\begin{aligned}
\Pr_u[u \in C_{\pi(2)} | \pi] &= \Pr_u[u \notin S_{\pi(1)} \wedge u \in S_{\pi(2)} | \pi] \\
&= \Pr_u[u \notin S_{\pi(1)} | \pi] \bullet \Pr_u[u \in S_{\pi(2)} | \pi] \\
&= \left(1 - \|\mathbf{y}_u^{\pi(1)}\|^2\right) \bullet \Pr_u\left[\mathbf{z}^{\pi(2)} \bullet \mathbf{g}_2 \geq \Phi^{-1}\left(1 - \frac{\|\mathbf{y}_u^{\pi(2)}\|^2}{(1 - \|\mathbf{y}^{\pi(1)}\|^2)_u}\right) | \pi\right] \\
&= \left(1 - \|\mathbf{y}_u^{\pi(1)}\|^2\right) \bullet \frac{\|\mathbf{y}_u^{\pi(2)}\|^2}{(1 - \|\mathbf{y}_u^{\pi(1)}\|^2)} \\
&= \|\mathbf{y}_u^{\pi(2)}\|^2.
\end{aligned}$$

Finally, since the events $\{u \in C_{\pi(1)} | \pi\}$, $\{u \in C_{\pi(2)} | \pi\}$, and $\{u \in C_{\pi(3)} | \pi\}$ are disjoint and exactly one of them occurs, i.e., every vertex $u \in V$ belongs to exactly one cluster in the output, we can conclude that:

$$\Pr[u \in C_{\pi(3)} \mid \pi = 1] \geq \Pr[u \in C_{\pi(1)} \mid \pi] \cdot \Pr[u \in C_{\pi(2)} \mid \pi = 1] \cdot \|\mathbf{y}_u^{\pi(1)}\|^2 \cdot \|\mathbf{y}_u^{\pi(2)}\|^2 = \|\mathbf{y}_u^{\pi(3)}\|^2.$$

In the above the last equality follows from Constraint (4). Unfixing the conditioning on $\pi$ by using the law of total probability concludes the proof. ◀

Our goal is to write an expression for the probability that an edge crosses between two different parts in the partition that Algorithm 1 outputs. Given a fixed pair of vertices $u, v \in V$ and $i = 1, 2, 3$, we denote by $t_i$ the inner product between $\mathbf{z}_u$ and $\mathbf{z}_v$. One should note that Constraint 3 in SDP 1 implies:

$$t_i = \frac{(\mathbf{y}_u^i - x_i \cdot \mathbf{y}_\varnothing) \cdot (\mathbf{y}_v^i - w_i \cdot \mathbf{y}_\varnothing)}{\sqrt{(x_i - x_i^2) \cdot (w_i - w_i^2)}} = \frac{\alpha_i - x_i w_i}{\sqrt{(x_i - x_i^2) \cdot (w_i - w_i^2)}}, \tag{9}$$

where $x_i = \|\mathbf{y}_u^i\|^2$ and $w_i = \|\mathbf{y}_v^i\|^2$ are the marginal values the SDP assigns to vertices $u$ and $v$, respectively, with respect to the $i^{\text{th}}$ cluster, and $\alpha_i = \mathbf{y}_u^i \cdot \mathbf{y}_v^i$ is the correlation the SDP assigns for both $u$ and $v$ being assigned to the $i^{\text{th}}$ cluster. The following lemma gives the desired expression. We require the following claim for its proof:

▶ Claim 10. Let $(X, Y)$ be a standard bi-variate Gaussian with correlation $t$. Then for every $q_1, q_2 \in [0, 1]$, we have $\Pr[X \geq \Phi^{-1}(1 - q_1), Y \geq \Phi^{-1}(1 - q_2)] = \Gamma_t(q_1, q_2)$.

▶ **Lemma 11.** *For every $u, v \in V$, let $\mathrm{A}_{u,v}$ be the event that Algorithm 1 separates $u$ and $v$:*

$$\Pr[\mathrm{A}_{u,v}] = 1 - \frac{1}{6} \sum_{\pi \in \Sigma_3} \Big[ \Gamma_{t_{\pi(1)}}(x_{\pi(1)}, w_{\pi(1)}) + \Gamma_{t_{\pi(1)}}(1 - x_{\pi(1)}, 1 - w_{\pi(1)}) + \Gamma_{t_{\pi(2)}}\left(1 - \frac{x^{\pi(2)}}{1 - x_{\pi(1)}}, 1 - \frac{w^{\pi(2)}}{1 - w_{\pi(1)}}\right) + \Gamma_{t_{\pi(2)}}\left(\frac{x^{\pi(2)}}{1 - x_{\pi(1)}}, \frac{w^{\pi(2)}}{1 - w_{\pi(1)}}\right) \Big].$$

The proof of Lemma 11 appears in the full version of the paper.

Our goal now is to lower bound the expected value of the output of Algorithm 1, before it is re-balanced (with a negligible loss) to ensure the size of each cluster is exactly $n/3$. As our analysis is performed edge-by-edge, i.e., for every edge we lower bound the ratio of the probability Algorithm 1 separates the edge to the contribution of this edge to the objective of SDP (1), we introduce the notions of a configuration and a feasible configuration.

A *configuration* is a vector $\mathbf{c} = (x_1, x_2, x_3, w_1, w_2, w_3, \alpha_1, \alpha_2, \alpha_3, t_1, t_2, t_3) \in \mathrm{R}^{12}$, such that for every $i = 1, 2, 3$: $x_i, w_i, \alpha_i \in [0, 1]$ and $t_i \in [-1, 1]$. We say that a configuration $\mathbf{c}$ is a *feasible configuration* if it can be realized by vectors in a feasible solution to SDP (as the following definition states).

▶ **Definition 12.** *A configuration $\mathbf{c} = (x_1, x_2, x_3, w_1, w_2, w_3, \alpha_1, \alpha_2, \alpha_3, t_1, t_2, t_3) \in [0, 1]^9 \times [-1, 1]^3$ is called a* feasible configuration *if there are vectors $\mathbf{y}_u^1, \mathbf{y}_u^2, \mathbf{y}_u^3, \mathbf{y}_v^1, \mathbf{y}_v^2, \mathbf{y}_v^3$ and $\mathbf{y}_\varnothing$ satisfying:*

1. *The vectors $\mathbf{y}_u^1, \mathbf{y}_u^2, \mathbf{y}_u^3, \mathbf{y}_v^1, \mathbf{y}_v^2, \mathbf{y}_v^3$ and $\mathbf{y}_\varnothing$ satisfy Constraints (2) to (6) in* SDP.
2. *$x_i = \|\mathbf{y}_u^i\|^2$, $w_i = \|\mathbf{y}_v^i\|^2$ and $\alpha_i = \mathbf{y}_u^i \cdot \mathbf{y}_v^i$, $\forall i = 1, 2, 3$.*
3. *$t_i = (\alpha_i - x_i w_i)/((x_i - x^2)(w_i - w^2))^{1/2}$, $\forall i = 1, 2, 3$.*

*The vectors $\mathbf{y}_u^1, \mathbf{y}_u^2, \mathbf{y}_u^3, \mathbf{y}_v^1, \mathbf{y}_v^2, \mathbf{y}_v^3$ and $\mathbf{y}_\varnothing$ are called a* realization *of $\mathbf{c}$. The set of all feasible configuration is denoted by $\mathrm{X}$.*

We note that there is some redundancy in the above definition. First, we can reduce the dimension of the configuration by two simply by substituting $x_3$ with $1 - x_1 - x_2$ and $w_3$ with $1 - w_1 - w_2$. Second, we can remove $t_1$, $t_2$ and $t_3$ (or $\alpha_1$, $\alpha_2$ and $\alpha_3$) since each $t_i$ (or $\alpha_i$) can be derived from all the parameters excluding $t_1$, $t_2$, and $t_3$ (or excluding $\alpha_1$, $\alpha_2$, and $\alpha_3$). In Lemma 21, we give a characterization of $\mathrm{X}$ after projecting out $t_1, t_2, t_3$. This description consists of linear constraints and we utilize the description for our computer-assisted proof. For simplicity of the analysis, we will keep all the parameters.

Let us now define two functions over $\mathrm{X}$. The first function $f$, given a feasible configuration $\mathbf{c} \in \mathrm{X}$, returns the probability that Algorithm 1 cuts an edge whose associated vectors are a realization of $\mathbf{c}$. Formally, following Lemma 11:

$$f(\mathbf{c}) = 1 - \frac{1}{6}\sum_{\pi\in\Sigma_3} \Big[\Gamma_{t_{\pi(1)}}\left(x_{\pi(1)}, w_{\pi(1)}\right) + \Gamma_{t_{\pi(1)}}\left(1-x_{\pi(1)}, 1-w_{\pi(1)}\right) \cdot \Gamma_{t_{\pi(2)}}\left(1-\frac{x_{\pi(2)}}{1-x_{\pi(1)}}, 1-\frac{w_{\pi(2)}}{1-w_{\pi(1)}}\right) + \Gamma_{t_{\pi(2)}}\left(\frac{x_{\pi(2)}}{1-x_{\pi(1)}}, \frac{w_{\pi(2)}}{1-w_{\pi(1)}}\right)\Big].$$

The second function $g$, given a feasible configuration $\mathbf{c} \in \mathrm{X}$, returns the contribution to the objective of SDP of an edge whose associated vectors are a realization of $\mathbf{c}$. Formally, following (1):

$$g(\mathbf{c}) = 1 - \alpha_1 - \alpha_2 - \alpha_3.$$

It is important to note that both $f$ and $g$ can be evaluated for every configuration $\mathbf{c} \in [0,1]^9 \times [-1,1]^3$ which might not be necessarily feasible. However, for such a (non feasible) configuration $\mathbf{c}$, $f(\mathbf{c})$ and $g(\mathbf{c})$ lose their "meaning".

To lower bound the value of the solution $\{C_1, C_2, C_3\}$ Algorithm 1 outputs, we introduce the following:

$$\mu = \inf_{\mathbf{c}\in\mathrm{X}} \frac{f(\mathbf{c})}{g(\mathbf{c})} \quad . \tag{10}$$

Clearly, from the above definition of $\mu$, the value of the output $\{C_1, C_2, C_3\}$ of Algorithm 1 is at least: $\mu \cdot \mathrm{OPT}_{\mathrm{SDP}} \geq \mu \cdot \mathrm{OPT}$ (where the inequality follows from Lemma 5 which states that SDP is a relaxation). Hence, if the output $\{C_1, C_2, C_3\}$ of Algorithm 1 was perfectly balanced, i.e., $|C_1| = |C_2| = |C_3| = n/3$, Algorithm 1 would achieve an approximation of $\mu$ to the Max-3-Section problem. In what follows we show that one can re-balance $\{C_1, C_2, C_3\}$ without a significant loss in the value of the solution. Thus, our goal is to lower bound $\mu$. For now, as formally proving the exact value of $\mu$ is a challenging task, we state the following conjecture that follows from numeric estimation of $\mu$.

**Conjecture 13.** $\mu \geq 0.8192$.

Assuming the above conjecture regarding $\mu$ (Conjecture 13), there are two things that are left in order to conclude the analysis of our algorithm. First, we show that if the solution $Y$ to SDP is independent (as in Definition 7 and Lemma 8 ) then with a suficiently high probability every cluster $C_i$ is close to the desired size of $n/3$. This gives rise to the following Definition 14 and Lemma 15. Second, we show that a solution $\{C_1, C_2, C_3\}$ that is close to being perfectly balanced can be eficiently re-balanced without a significant loss in its value. The latter is summarized in Lemma 16.

**Definition 14.** *A partition* $\{C_1, C_2, C_3\}$ *of a graph on* $n$ *nodes is* $\epsilon$-unbalanced *if for every* $i = 1, 2, 3$*:*

$$\frac{n}{3}(1 - \epsilon) \leq |C_i| \leq \frac{n}{3}(1 + \epsilon).$$

**Lemma 15.** *Let* $Y$ *be a* $\frac{1}{t}$*-independent solution to* SDP *where* $t = \Omega(\epsilon^{-18})$*, and* $\{C_1, C_2, C_3\}$ *be the partition that Algorithm 1 outputs on* $Y$*. Then for every* $i = 1, 2, 3$ *it holds that:*

$$Pr\left[||C_i| - n/3| \geq \frac{\epsilon n}{3}\right] \leq \epsilon.$$

Next, we show that such unbalanced partition can be balanced without a large loss in the objective. That is, we present an algorithm that given a $\epsilon$-unbalanced partition, finds in polynomial time a balanced partition with small loss in the objective, in expectation.

**Lemma 16.** *There is a polynomial-time algorithm that given a* $\epsilon$*-unbalanced partition* $\{C_1, C_2, C_3\}$ *with value* $\Delta = |\delta(C_1, C_2)| + |\delta(C_2, C_3)| + |\delta(C_1, C_3)|$ *finds a balanced partition* $\{C_1', C_2', C_3'\}$ *with expected value* $E[\Delta'] \geq (1 - 2\epsilon)\Delta$*.*

The proofs of Lemma 15 and Lemma 16 appear in the full version of the paper. We combine these lemmas and prove the following result.

**Theorem 17.** *For every constant* $\varepsilon > 0$*, there exists a polynomial-time approximation algorithm for* Max-3-Section*, that runs in time* $n^{O(poly(\epsilon^{-1}))}$*, achieving an approximation of* $(1 - 2\epsilon)(\mu - O(\varepsilon))$*.*

**Proof.** Let $\varepsilon > 0$ be a constant, and $t$ an integer satisfying $t = \Omega(\varepsilon^{-18})$. Lemma 8 shows that we can compute in polynomial time a solution $Y$ to SDP that is $\frac{1}{t}$-independent with only an additive loss of $1/t$ in the objective. We repeatedly apply Algorithm 1 to round the above $Y$ until we obtain a solution $\{C_1, C_2, C_3\}$ that is $\varepsilon$-unbalanced, and then apply Lemma 16 to re-balance it an obtain our final output.

Let us now analyze the approximation guarantee of the above algorithm. First, It follows form Lemma 15 and a simple union bound over the three clusters that with a probability of at least $1 - 3\varepsilon$, applying Algorithm 1 to round the above solution $Y$ yields a clustering $\{C_1, C_2, C_3\}$ that is $\varepsilon$-unbalanced (as in Definition 14). Let us denote by $\mathrm{A}$ the event that $\{C_1, C_2, C_3\}$ is $\varepsilon$-unbalanced. Hence, $\Pr[\mathrm{A}] \geq 1 - 3\varepsilon$ and $\Pr[\bar{\mathrm{A}}] \leq 3\varepsilon$. Moreover, as before, let us denote by $\Delta$ the value of the solution $\{C_1, C_2, C_3\}$. Thus, the expected value of this solution *conditioned* on it being $\varepsilon$-unbalanced is at least:

$$E[\Delta|\mathrm{A}] = \frac{E[\Delta] - \Pr[\bar{\mathrm{A}}] \cdot E[\Delta|\bar{\mathrm{A}}]}{\Pr[\mathrm{A}]} \geq \mu \cdot \mathrm{OPT}_{\mathrm{SDP}-\mathrm{H}} - \frac{9\varepsilon}{2}\mathrm{OPT}.$$

The inequality follows from the facts that: (1) $E[\Delta|\bar{\mathrm{A}}] \leq (3/2) \cdot \mathrm{OPT}$ (since $\Delta \leq m$ and $\mathrm{OPT} \geq (2/3)m$); (2) $E[\Delta] \geq \mu \cdot \mathrm{OPT}_{\mathrm{SDP}-\mathrm{H}}$ (definition of $\mu$ (10)); and (3) $\Pr[\mathrm{A}] \leq 1$, $\Pr[\bar{\mathrm{A}}] \leq 3\varepsilon$. We note that $\mathrm{OPT}_{\mathrm{SDP}-\mathrm{H}} \geq \mathrm{OPT}_{\mathrm{SDP}} - \varepsilon^{18} \geq \mathrm{OPT} - \varepsilon^{18}$. Hence,

$$E[\Delta|\mathrm{A}] \geq \mathrm{OPT}(\mu - O(\varepsilon)).$$

Applying Lemma 16 concludes the proof. ◻

Moreover, in Observation 34 in the full version of the paper we present a configuration $\mathbf{c}$ that has a ratio of $\frac{f(\mathbf{c})}{g(\mathbf{c})} = 0.8192$, hence that is an upper bound on $\mu$.

### 4.1 Towards Estimating $\mu$ via a Computer Assisted Proof

For our computer assisted proof, we consider a slightly different version of $\mu$ which speeds up our code. Consider the following, for some fixed $\delta' > 0$:

$$\mu' = \inf_{\mathbf{c} \in X, g(\mathbf{c}) \geq \delta'} \frac{f(\mathbf{c})}{g(\mathbf{c})} \quad . \tag{11}$$

The following lemma shows the loss we incur when using $\mu'$ rather than $\mu$ is bounded.

**Lemma 18.** *Let $\{C_1, C_2, C_3\}$ be the output of Algorithm 1 when run on a solution $Y$ to* SDP *1 with objective value* $\mathrm{SDP}_{\mathrm{VAL}}$. *Then we have that*

$$\mathrm{E}[|\delta(C_1, C_2)| + |\delta(C_1, C_3)| + |\delta(C_2, C_3)|] \geq \left(1 - \frac{\delta'}{2(1-\delta')}\right) \mu' \cdot \mathrm{SDP}_{\mathrm{VAL}}.$$

The following theorem summarizes the approximation guarantee when $\mu'$ is used instead of $\mu$.

**Theorem 19.** *For any constant $\epsilon > 0$, there is an algorithm that outputs a partition of the vertex set $\{C_1, C_2, C_3\}$ with $|C_1| = |C_2| = |C_3|$ satisfying,*

$$\mathrm{E}[|\delta(C_1, C_2)| + |\delta(C_1, C_3)| + |\delta(C_2, C_3)|] \geq (1-2\epsilon)\left(1 - \frac{\delta'}{2(1-\delta')}\right) \mu' (\mathrm{OPT}_{\mathrm{SDP}} - O(\epsilon^{3/2}))$$

*with an expected run-time of $n^{O(1/\epsilon)}$.*

**Proof.** First we use Lemma 8 to get a $\frac{1}{t}$-independent solution $Y$ for $t = \Omega(\epsilon^{-18})$. Let $\mathrm{OPT}_{\mathrm{SDP}-H}$ denote the objective value of this solution. Next we run Algorithm 1 on $Y$ and repeat until it outputs sets $C_1', C_2', C_3'$ that are $\epsilon$-unbalanced. We note that by Lemma 15 $(C_1', C_2', C_3')$ is *not* $\epsilon$-unbalanced with probability at most $3\epsilon$, so we run Algorithm 1 at most $1/3\epsilon$ times in expectation. Since $C_1', C_2', C_3'$ are $\epsilon$-unbalanced, applying the random shifting of Lemma 16 to $C_1', C_2', C_3'$ we get $C_1, C_2, C_3$ satisfying

$$\mathrm{E}[|\delta(C_1, C_2)| + |\delta(C_1, C_3)| + |\delta(C_2, C_3)|] \geq (1-2\epsilon)\mathrm{E}[|\delta(C_1', C_2')| + |\delta(C_1', C_3')| + |\delta(C_2', C_3')|],$$

From Lemma 18 we have,

$$\mathrm{E}[|\delta(C_1, C_2)| + |\delta(C_1, C_3)| + |\delta(C_2, C_3)|] \geq (1-2\epsilon)\left(1 - \frac{\delta'}{2(1-\delta')}\right) \mu' \mathrm{OPT}_{\mathrm{SDP}-H}.$$

Then applying second item of Lemma 8 gives us that $\mathrm{OPT}_{\mathrm{SDP}-H} \geq \mathrm{OPT}_{\mathrm{SDP}} - O(\epsilon^{3/2})$ since we set $t = \Omega(\epsilon^{-18})$.

## 5 Computer Assisted Proof

The goal of this section is to lower bound $\mu'$. We use a branch and bound procedure to lower $\mu'$ and which gives a guarantee for the approximation factor of our algorithm (Theorem 19) . We recall the definition of $\mu'$

$$\mu' = \inf_{\mathbf{c} \in X, g(\mathbf{c}) \geq \delta'} \frac{f(\mathbf{c})}{g(\mathbf{c})} \quad . \tag{12}$$

The following claim gives a simple, yet useful, bound on the probability that our rounding algorithm will separate two vertices in the graph.

□ Claim 20. Let $u, v \in V$, let $x_i = \|\mathbf{y}_u^i\|^2$ and $w_i = \|\mathbf{y}_v^i\|^2$ for $i = 1, 2, 3$ and $\mathbf{c}$ be a configuration corresponding to this pair. Then it holds that

$$f(\mathbf{c}) \geq \frac{|x_1 - w_1| + |x_2 - w_2| + |x_3 - w_3|}{2}.$$

We characterize the configuration space $\mathrm{X}$ which we will consider in the computer assisted proof.

□ **Lemma 21.** *Let* $\mathbf{c} = (x_1, x_2, x_3, w_1, w_2, w_3, \alpha_1, \alpha_2, \alpha_3, t_1, t_2, t_3) \in \mathrm{X}$. *Then* $\mathbf{c}$ *satisfies the following:*

1. $x_1 + x_2 + x_3 = w_1 + w_2 + w_3 = 1$.
2. $0 \leq \alpha_i \leq \min\{x_i, w_i\}$ *for all* $i = 1, 2, 3$.
3. $\max\{0, x_3 - \alpha_3 + \alpha_1 - w_1, w_2 - \alpha_2 + \alpha_1 - x_1\} \leq \min\{w_2 - \alpha_2, x_3 - \alpha_3, x_2 + x_3 - w_1 + \alpha_1 - \alpha_2 - \alpha_3\}$.
4. $t_i = \frac{\sqrt{\alpha_i - x - w}}{(x_i - x^2)_i (w - w^2)_i}$ *for* $i \in [3]$.

We define the following two polytopes which we will consider when verifying bounding $\mu'$. These polytopes help us to speed up the branch and bound procedure.

1. $\Sigma = \{(x_1, x_2, x_3, w_1, w_2, w_3, \alpha_1, \alpha_2, \alpha_3, t_1, t_2, t_3) | x_1 \leq \min(x_2, w_1, w_2, w_3, x_3), x_2 \leq x_3\}$.
2. $\mathrm{E} = \{\mathbf{c} = (x_1, x_2, x_3, w_1, w_2, w_3, \alpha_1, \alpha_2, \alpha_3, t_1, t_2, t_3)\}$ such that

$$\sum_{i=1}^{3} \frac{|x_i - w_i|}{2} \leq \rho g(\mathbf{c}), g(\mathbf{c}) \geq \delta'\}.$$

The following claim shows why we can restrict to configurations in $\Sigma$ and follows from the symmetry of $f, g$.

□ Claim 22. Let $\mathbf{c} \in \mathrm{X} - \Sigma$. Then there exists $\mathbf{c}' \in \mathrm{X} \cap \Sigma$ such that $f(\mathbf{c}) = f(\mathbf{c}')$ and $g(\mathbf{c}) = g(\mathbf{c}')$.

Then Claim 22 and Claim 20 imply the following.

□ Claim 23. If $\inf_{\mathbf{c} \in \mathrm{X} \cap \Sigma \cap \mathrm{E}} \frac{f(\mathbf{c})}{g(\mathbf{c})} \geq \rho$ then $\mu' \geq \rho$.

Proof. By Claim 22 we get the following: $\mu' = \inf_{\mathbf{c} \in \mathrm{X}, g(\mathbf{c}) \geq \delta'} \frac{f(\mathbf{c})}{g(\mathbf{c})} = \inf_{\mathbf{c} \in \mathrm{X} \cap \Sigma, g(\mathbf{c}) \geq \delta'} \frac{f(\mathbf{c})}{g(\mathbf{c})}$. Assume for contradiction that $\mu' = \min_{\mathbf{c} \in \mathrm{X} \cap \Sigma, g(\mathbf{c}) \geq \delta'} \frac{f(\mathbf{c})}{g(\mathbf{c})} < \rho$. Then $\exists \mathbf{c}' \in \mathrm{X} \cap \Sigma$ with $g(\mathbf{c}') \geq \delta'$ and $\frac{f(\mathbf{c}')}{g(\mathbf{c}')} < \rho$. We have that $\mathbf{c}' \notin \mathrm{E}$ otherwise $g(\mathbf{c}') \geq \rho$ by the assumption of the lemma. Then by definition of $\mathrm{E}$, $\sum_{i=1}^{3} \frac{|x_i - w_i|}{2} > \rho g(\mathbf{c}')$, but this is a contradiction by Claim 20 since $f(\mathbf{c}') > \sum_{i=1}^{3} \frac{|x_i - w_i|}{2} > \rho g(\mathbf{c}')$. □

Thus our goal for the computed assisted proof is to show $\inf_{\mathbf{c} \in \mathrm{X} \cap \Sigma \cap \mathrm{E}} \frac{f(\mathbf{c})}{g(\mathbf{c})} \geq \rho$ for some value of $\rho$. Given 12 intervals $(I_1, \ldots, I_{12})$ we define the following polytopes to divide our feasible region into hypercubes. Consider the following polytope,

$$\Pi(I_1, \ldots, I_{12}) = (I_1 \times I_2 \ldots \times I_{12}) \cap \mathrm{X}.$$

Note that intervals $I_j$ correspond to possible values of $x_j$ for $j \in [3]$, intervals $I_3, I_4, I_5$ correspond to values of $w_1, w_2, w_3$, intervals $I_7, I_8, I_9$ correspond to values of $\alpha_1, \alpha_2, \alpha_3$, and intervals $I_{10}, I_{11}, I_{12}$ correspond to $t_1, t_2, t_3$. Our computer assisted proof enumerates $I_1, \ldots, I_{12}$ so that the union of all $\Pi(I_1, \ldots, I_{12}) \cap \Sigma \cap \mathrm{E}$ covers $\mathrm{X} \cap \Sigma \cap \mathrm{E}$. For each $I_1, \ldots, I_{12}$ we show one of the following three:

1. $\Pi(I_1, \ldots, I_{12}) \cap \Sigma \cap \mathrm{E} = \emptyset$.
2. $\inf_{\mathbf{c} \in \Pi(I_1,\ldots,I_{12}) \cap \Sigma \cap \mathrm{E}} \frac{f(\mathbf{c})}{g(\mathbf{c})} \geq \rho$.
3. Divide $(I_1, \ldots, I_{12})$ into a collection $\mathrm{Y}$ whose union equals $(I_1, \ldots, I_{12})$ so that each $(I_1', \ldots, I_{12}') \in \mathrm{Y}$ satisfies one of the first two items.

which implies the hypothesis of Claim 23. The third item is the branching step and the first two items are how we eliminate branches. For our computer assisted proof we will only consider $x_1, x_2, w_1, w_2, t_1, t_2, t_3$ as independent variables. The remaining variables $w_3, x_3, \alpha_1, \alpha_2, \alpha_3$ will always take values $w_3 = 1 - w_1 - w_2$, $x_3 = 1 - x_1 - x_2$, $\alpha_i = x_i w_i + t_i \sqrt{(1 - x_i^2)(1 - w_i^2)}$ for $i \in [3]$. Our algorithm runs in stages. In the first stage we use an LP to eliminate hypercubes. The first stage works as follows,

1. Enumerate all $\mathrm{I} = (I_1, \ldots, I_{12})$ such that $|I_j| = \eta_1$ for $j \in \{1, 2, 4, 5\}$ and $|I_j| = \eta_2$ for $j = 10, 11, 12$ that the union of all $(I_1, I_2, I_3, I_4, I_{10}, I_{11}, I_{12})$ covers the region $[0, 1]^4 \times [-1, 1]^3$. Note that intervals $I_5, I_6, I_7$ can be determined by the bounds on the other intervals, $I_j$ such that $j \notin \{5, 6, 7\}$. This follows since bounds on $x_i, w_i, t_i$ imply bounds on $\alpha_i$ because $\alpha_i = t_i \sqrt{(1 - x^2)(1 - w^2)} + x_i w_i$. Similarly, $I_3, I_6$ are determined by $I_1, I_2$ and $I_4, I_5$ respectively since $x_3 = 1 - x_1 - x_2$ and $w_3 = 1 - w_1 - w_2$.
2. For each hypercube $\mathrm{I}$ enumerated in the previous step, check that $\Pi(\mathrm{I}) \cap \mathrm{E} \cap \Sigma$ contains a feasible point by solving an LP. If yes, save $\mathrm{I}$ for further processing.

### Partial Derivatives

A crucial ingredient of the branch and bound procedure is to obtain a lower bound on the function $f(\mathbf{c})$ for any configuration $\mathbf{c} \in \mathrm{I}$ in any cube. This we do by computing $f(\mathbf{c}')$ for some well chosen $\mathbf{c}' \in \mathrm{I}$ and then using bounds on the partial derivatives to infer a bound $f(\mathbf{c})$ for all other $\mathbf{c} \in \mathrm{I}$. A tight bound on partial derivatives ensures a smaller branch and bound tree. We detail the partial derivatives and bounds thus obtained now. Previously, we defined the notion of configuration and the function $f$ as a function of 12 variables. Since we only consider $(x_1, x_2, w_1, w_2, t_1, t_2, t_3)$ as variables we have $\frac{\partial f}{\partial x_3} = \frac{\partial f}{\partial w_3} = \frac{\partial f}{\partial \alpha_i} = 0$ and $f, g$ are functions of 7 independent variables. We recall the definition of $f$

$$f(x, w, \alpha, t) = 1 - \frac{1}{6} \sum_{\pi \in \Sigma_3} \Big[ \Gamma_{t_{\pi(1)}}\left(x_{\pi(1)}, w_{\pi(1)}\right) + \Gamma_{t_{\pi(1)}}\left(1 - x_{\pi(1)}, 1 - w_{\pi(1)}\right) \cdot \Big( \Gamma_{t_{\pi(2)}}\left(1 - \frac{x^{\pi(2)}}{1 - x_{\pi(1)}}, 1 - \frac{w^{\pi(2)}}{1 - w_{\pi(1)}}\right) + \Gamma_{t_{\pi(2)}}\left(\frac{x^{\pi(2)}}{1 - x_{\pi(1)}}, \frac{w^{\pi(2)}}{1 - w_{\pi(1)}}\right) \Big) \Big].$$

Moreover, for our use we can assume that the domain of $f$ is $0 \leq x_1 + x_2 \leq 1$, $0 \leq w_1 + w_2 \leq 1$, $\alpha_1, \alpha_2, \alpha_3 \in [0, 1]$ and $t_1, t_2, t_3 \in [-1, 1]$. In the following, we are going to prove bounds on the partial derivatives of $f(x, w, \alpha, t)$ with respect to $x_i$, $w_i$ and $t_i$.

**Lemma 24.** *For each $(x, w, \alpha, t)$ in the domain of configuration, the following bounds on the partial derivatives of $f$ hold:*

1. $\frac{\partial}{\partial t_i} f(x, w, \alpha, t) < 0$, *for* $i = 1, 2, 3$.
2. $\left|\frac{\partial}{\partial x_i} f(x, w, \alpha, t)\right| \leq \frac{5}{3}(1 - x_{3-i})$, *for* $i = 1, 2$.
3. $\left|\frac{\partial}{\partial w_i} f(x, w, \alpha, t)\right| \leq \frac{5}{3}(1 - w_{3-i})$, *for* $i = 1, 2$.

The proof of the lemma is technical and appears in the full version of the paper. We now have the following claim that gives a lower bound on all configurations in the hypercube as compared to the point $\mathbf{m}$ in it.

Claim 25. Let $\mathrm{I}$ be a hypercube and $\Theta \subseteq \Pi(\mathrm{I}) \cap \Sigma \cap \mathrm{E}$. Let $\mathbf{m} \in \mathrm{R}^{12}$ be a point where the coordinates corresponding to $x_1, x_2, w_1, w_2$ are the midpoints $l_1, l_2, l_4, l_5$ respectively and the last 3 coordinates corresponding are $\overline{t_1}, \overline{t_2}, \overline{t_3}$ which are the upper bounds of $l_{10}, l_{11}, l_{12}$. Then the following holds,

$$\min_{\mathbf{c}=(x_1,\ldots,t_1,t_2,t_3)\in\Theta} f(x_1, x_2, \ldots, \overline{t_1}, \overline{t_2}, \overline{t_3}) \geq f(\mathbf{m}) - \frac{1}{6} \sum_{z\in\{x,w\}} \sum_{i=1}^{2} (5 - (1 - z_{3-i}))(z_i - \underline{z_i}).$$

**Lemma 26.** *Let $\mathrm{I}$, $\mathbf{m}$, $\Theta$ be defined as in Claim 25. Then $\min_{\mathbf{c}\in\Theta} \frac{f(\mathbf{c})}{g(\mathbf{c})} \geq \rho$ if,*

$$f(\mathbf{m}) - \frac{1}{6} \sum_{z\in\{x,w\}} \sum_{i=1}^{2} (5 - (1 - z_{3-i}))(z_i - \underline{z_i}) \geq \rho \max_{\mathbf{c}\in\Theta} g(\mathbf{c})$$

*where $\overline{z_i}, \underline{z_i}$ for $z \in \{w, x\}$ are the upper and lower bounds given by their corresponding interval in $\mathrm{I}$.*

**Proof.** We use the fact that $f$ is non-increasing in $t_i$ and Claim 25 to get,

$$\begin{aligned}\min_{\mathbf{c}\in\Theta} f(\mathbf{c}) &\geq \min_{\mathbf{c}=(x_1,x_2,\ldots,t_1,t_2,t_3)\in\Theta} f(x_1, x_2, \ldots, \overline{t_1}, \overline{t_2}, \overline{t_3}) \\ &\geq f(\mathbf{m}) - \frac{1}{6} \sum_{z\in\{x,w\}} \sum_{i=1}^{2} (5 - (1 - z_{3-i}))(z_i - \underline{z_i})\end{aligned}$$

The first inequality follows since $f$ is non-increasing in $t, \alpha$ by Lemma 24. The third inequality follows by Claim 25. Thus by the inequality in the lemma and the relation above, we have shown: $\min_{\mathbf{c}\in\Theta} f(\mathbf{c}) \geq \rho \max_{\mathbf{c}\in\Theta} g(\mathbf{c})$. ∎

Now we describe the final stage of the experiment which eliminates all remaining cubes from the first stage.

1. For all remaining hyper cubes $\mathrm{I}$ from stage 1 run the following steps until all cubes are eliminated.
2. Split the intervals $l_1, l_2, l_4, l_5$ corresponding to $x_1, x_2, w_1, w_2$ into halves to get 16 smaller sub-hypercubes $(l_1', \ldots, l_9', l_{10}, l_{11}, l_{12})$. We note that this also tightens the intervals corresponding to $x_3, w_3, \alpha_1, \alpha_2, \alpha_3$. Check if each smaller hypercube has a feasible point in $\Pi(l_1, \ldots, l', l_{10}, l_{11}, l_{12}) \cap \Sigma \cap \mathrm{E}$. If yes, then the sub-hypercube can be eliminated.
3. Otherwise, verify if the inequality in 26 holds. If yes, the sub-hypercube can be eliminated.
4. Otherwise, split the $t_1, t_2, t_3$ intervals into halves to get 8 sub-hypercubes $\mathrm{I}' = (l_1', \ldots, l_9', l_{10}'', l_{11}'', l_{12}'')$. This also tightens the intervals for $\alpha_1, \alpha_2, \alpha_3$. Check if each smaller hypercube $\mathrm{I}'$ has a feasible point in $\Pi(\mathrm{I}') \cap \Sigma \cap \mathrm{E}$. If yes, then the sub-hypercube can be eliminated.
5. Otherwise, split the sub-hyper cube into 16 smaller hyper-cubes using Step 2. Repeat Steps 2-5 until all sub cubes are eliminated.

We ran this branch and bound procedure with $\rho = 0.80$, $\delta' = 0.01$ giving a final approximation of 0.795.

## 6 Max-k-Section

Our algorithm for Max-3-Section can be generalized for larger number of sections in the following way. For a natural number $k \geq 4$, one can write a similar SDP formulation, where each node $v \in V$ has $k$ vectors $\mathbf{y}_v^1, \ldots, \mathbf{y}_v^k$, with similar constraints and objective for the SDP

for Max-3-Section: maximize $\sum_{\{u,v\}\in E} \bullet(1 - \sum_{i=1}^{k} \mathbf{y}_u^i \bullet \mathbf{y}_v^i)$. The rounding algorithm will work in a similar way. For example, we consider $k = 4$. We draw a random permutation $\pi$ in $\Sigma^4$ and three random gaussian vectors $\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3$ with coordinates independently distributed by $N(0, 1)$. Then, we define $S_{\pi(1)}$ and $S_{\pi(2)}$ in the same way like in the algorithm for Max-3-Section. Next, we define

$$S_{\pi(3)} = \left\{ u \in V : \mathbf{z}_u^{\pi(3)} \bullet \mathbf{g}_3 \geq \Phi^{-1}\left(1 - \frac{\mathbf{y}_u^{\pi(3)} \cdot \mathbf{y}_u^{\pi(1)}}{1 - \left(\mathbf{y}_u^{\pi(1)}\right)^2 - \left(\mathbf{y}_u^{\pi(2)}\right)^2}\right)\right\}$$

and the four clusters in the output will be $C_{\pi(1)} = S_{\pi(1)}$, $C_{\pi(2)} = S_{\pi(2)} \setminus S_{\pi(1)}$, $C_{\pi(3)} = S_{\pi(3)} \setminus (S_{\pi(2)} \cup S_{\pi(1)})$ and $C_{\pi(4)} = V \setminus (S_{\pi(1)} \cup S_{\pi(2)} \cup S_{\pi(3)})$. Then, for any constant $k$, we can claim that with high probability the solution is concentrated and can be re-balanced, similarly to Lemma 15 and Lemma 16.

Our goal now is to bound the approximation factors that these algorithms achieve, for each $k$. As we discussed in the previous sections, computing the worst approximation guarantee, or even bounding it analytically is not an easy task. For $k = 3$, we used the branch and bound algorithm and presented a lower bound on the approximation ratio. For larger values of $k$, the computer-assisted proof method becomes computationally harder. However, one possible approach is to try and give a numerical estimation for the approximation factor. We will do that in the following way: for each $k$, one can write an optimization problem over $k^2$ variables that represent the inner products $\mathbf{y}_u^i \bullet \mathbf{y}_v$, for each $i, j \in [k]$, or as we denoted before, a feasible configuration. Then, we wish to minimize the ratio of the probability that $u, v$ are separated and the contribution of the edge $\{u, v\}$ to the SDP, which is $1 - \sum_{i=1}^{k} \mathbf{y}_u^i \bullet \mathbf{y}_v$. To solve that optimization problem, we use Matlab and the *fmincon* functionality which can find a local minimum for this optimization problem, but only a local minimum. Therefore, we repeat the experiment for numerous random starting points in the feasible region. The results of the numerical estimations of the approximation for $k = 3, 4, 5$ are presented in Conjecture 3.

We note that given a configuration of vectors $\mathbf{y}_u^1, \mathbf{y}_u^2, \mathbf{y}_u^3, \mathbf{y}_v^1, \mathbf{y}_v^2, \mathbf{y}_v^3$ that has a ratio of $\rho$ between the separation probability and the contribution of the edge $(u, v)$ to the SDP solution for Max-3-section, one can construct a configuration for max-4-section by adding two zero vectors $\mathbf{y}_u^4, \mathbf{y}_v^4$. That configuration will have the same contribution for the SDP for max-4-section, and the separation probability will also be the same, even though the algorithm admits four sections. Therefore, in that way of analysis, the approximation ratio of our algorithm can only decrease as $k$ increases. However, our numerical estimations show that for $k \leq 5$, the approximation is not worse than 0.8192. In addition, we note that the simple algorithm that returns a random balanced $k$-partition achieves a $1 - \frac{1}{k}$ approximation, hence for $k = 3, 4, 5$ our algorithm surpasses it.

## References


1. Gunnar Andersson. An approximation algorithm for max p-section. In *STACS 99: 16th Annual Symposium on Theoretical Aspects of Computer Science Trier, Germany, March 4–6, 1999 Proceedings*, pages 237–247. Springer, 2002.
2. Per Austrin, Siavosh Benabbas, and Konstantinos Georgiou. Better balance by being biased: A 0.8776-approximation for max bisection. *ACM Trans. Algorithms*, 13(1):2:1–2:27, 2016.
3. Etienne De Klerk, Dmitrii Pasechnik, Renata Sotirov, and Cristian Dobre. On semidefinite programming relaxations of maximum k-section. *Mathematical programming*, 136(2):253–278, 2012.

4 Etienne de Klerk, Dmitrii V Pasechnik, and Joost P Warners. On approximate graph colouring and max-k-cut algorithms based on the $\vartheta$-function. *Journal of Combinatorial Optimization*, 8:267–294, 2004.

5 Uriel Feige and Michael Langberg. The rpr2 rounding technique for semidefinite programs. *Journal of Algorithms*, 60(1):1–23, 2006.

6 Alan Frieze and Mark Jerrum. Improved approximation algorithms for max k-cut and max bisection. *Algorithmica*, 18(1):67–81, 1997.

7 Daya Ram Gaur, Ramesh Krishnamurti, and Rajeev Kohli. The capacitated max k-cut problem. *Mathematical Programming*, 115:65–72, 2008.

8 Michel X Goemans and David Williamson. Approximation algorithms for max-3-cut and other problems via complex semidefinite programming. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 443–452, 2001.

9 Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

10 Eran Halperin and Uri Zwick. A unified framework for obtaining improved approximation algorithms for maximum graph bisection problems. *Random Structures & Algorithms*, 20(3):382–402, 2002.

11 Richard M Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 1972.

12 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, 2007.

13 Ai-fan Ling. Approximation algorithms for max 3-section using complex semidefinite programming relaxation. In *Combinatorial Optimization and Applications: Third International Conference, COCOA 2009, Huangshan, China, June 10-12, 2009. Proceedings 3*, pages 219–230. Springer, 2009.

14 Alantha Newman. Complex semidefinite programming and max-k-cut. In *SIAM Symposium on Simplicity in Algorithms*, 2018.

15 Prasad Raghavendra and Ning Tan. Approximating csps with global cardinality constraints using SDP hierarchies. In *SODA*, pages 373–387. SIAM, 2012.

16 Yinyu Ye. A .699-approximation algorithm for max-bisection. *Mathematical Programming*, 90(1):101–111, March 2001.