Fast SDP Algorithms for Constraint Satisfaction Problems[*]

David Steurer[†]

Abstract

The class of constraint satisfactions problems (CSPs) captures many fundamental combinatorial optimization problems such as Max Cut, Max q-Cut, Unique Games, and Max k-Sat. Recently, Raghavendra (STOC'08) identified a simple semidefinite programming relaxation that gives the best possible approximation for any CSP, assuming the Unique Games Conjecture. Raghavendra and Steurer (FOCS'09) showed that, independent of the truth of the Unique Games Conjecture, the integrality gap of this relaxation cannot be improved even by adding a large class of valid inequalities.

We present an algorithm that finds an approximately optimal solution to this relaxation in near-linear time. Combining this algorithm with a rounding scheme of Raghavendra and Steurer (FOCS'09) leads to an approximation algorithm for any CSP that runs in near-linear time and has an approximation guarantee that matches the integrality gap, which is optimal assuming the Unique Games Conjecture.

1 Introduction

In a constraint satisfaction problem (CSP) instance, we are given a finite alphabet Σ and a set of variables V, and the goal is maximize an objective function over all assignments of alphabet symbols to the variable set V. The objective function is given as a convex combination of bounded, *local payoff functions* $\phi \colon \Sigma^S \to [-1,1]$, where $S \subseteq V$ is a subset of variables. The maximum number of variables that one of the local payoff function depends on is called the *arity* of the problem instance.

A constraint satisfaction problem Π is specified by an alphabet and a finite list of admissible payoff functions over that alphabet. The constraint satisfaction problem consists of all CSP instances that only use admissible payoff functions.

Many classical combinatorial optimization problems are constraints satisfaction problems. Prominent examples are Max Cut, Max 2-Sat, Max q-Cut, Unique Games, and Max 3-Sat.

Goemans and Williamson [GW95] introduced semidefinite programming (SDP) as a technique for approximating CSPs: they showed that a natural SDP relaxation yields a 0.878 approximation for Max Cut. In the following years, many SDP-based approximation algorithm have been developed for various CSPs [FJ97, FG95, KZ97, Zwi98a, Zwi98b, Zwi99, TSSW00, HZ01, MM01, LLZ02, CW04, GW04, Has05, CMM06a, CMM06b, CMM07]. It is remarkable that for many of these problems no other technique is known that achieves equally good approximations.

Recently, Raghavendra [Rag08] identified a simple semidefinite programming relaxation that gives the best possible approximation for any CSP, assuming Khot's Unique Games Conjecture [Kho02]. Raghavendra and Steurer [RS09b] showed that independent of the truth of the Unique Games Conjecture, the integrality gap of this relaxation cannot be improved for any CSP even by adding a large class of valid inequalities. (For Max Cut and Sparsest Cut, similar results have independently been obtained by Khot and Saket [KS09].)

We present an algorithm that finds an approximate solution to this relaxation in near-linear time. Combining this algorithm with a rounding scheme of Raghavendra and Steurer [RS09a] yields an approximation algorithm for any CSP that runs in near-linear time and has an approximation guarantee that matches the integrality gap. Assuming the Unique Games Conjecture, any better approximation is NP-hard.

Our algorithm is based on a framework of Arora and Kale [AK07] for solving semidefinite programs using the Matrix Multiplicative Weights method. In this framework, the crucial step is to design an efficient *width-bounded separation oracle.* We show how to implement this oracle by solving a sequence of *local linear programs.* We also generalize the framework of Arora and Kale in order to deal with "irregular" instances more efficiently. In particular, this generalization allows us to approximate Max Cut on non-regular graphs in near-linear time without reducing the instance to a regular graph beforehand [Tre09].

1.1 Results. Our main result is a near-linear time algorithm that given a CSP instance $\mathfrak{I}$, computes an approximately optimal SDP solution for a CSP instance $\mathfrak{I}'$ that approximates $\mathfrak{I}$. Both kinds of approximations can be made arbitrarily good at the cost of increasing the running time by a constant factor. (The symbol $\mathfrak{I}$ is a capital "i" in Fraktur font and is intended to be pronounced as "i".)

[*]Supported by NSF grants 0830673, 0832797, 528414.

[†]Computer Science Department, Princeton University. Part of this work was done while visiting MSR New England.

The following theorem is a more precise statement of our main result.

Theorem 1.1. *Let $\Im$ be a CSP instance on n variables with $m \geqslant n$ local payoff functions, alphabet size q, and arity k. Suppose the SDP value of $\Im$ is at least α. Then for every $\varepsilon > 0$, we can compute in time $m \cdot \mathrm{poly}(k^q/\varepsilon) \cdot \mathrm{polylog}\, n$ an SDP solution of value at least $\alpha - \varepsilon$ that is feasible for a CSP instance $\Im'$ obtained from $\Im$ by discarding at most an ε fraction of local payoff functions.*

We remark that the $\mathrm{polylog}\, n$ term in the running time can be improved to $O(\log n)^2$. In this extended abstract, we prove a slightly worse bound. We refer to the full version of this paper for a proof of the $O(\log n)^2$ bound.

We obtain approximation algorithms for CSPs by combining the previous theorem with a generic rounding algorithm for CSPs [RS09a]. Since this rounding algorithm also runs in near-linear time, the running time of the resulting approximation algorithm is still near-linear.

Let Π be a CSP (e.g., Max Cut, Max k-Sat, or Unique Games). Suppose k and q are the arity and the alphabet size of Π.

For a CSP instance $\Im$, we denote by $\mathrm{opt}(\Im)$ the value of an optimal assignment for $\Im$ and we denote by $\mathrm{sdp}(\Im)$ the value of an optimal SDP solution for $\Im$. (See §2.3 for the SDP relaxation.) Following Raghavendra [Rag08], we define the *integrality gap curve* of Π as

$$\mathrm{gap}(\Pi;\alpha) \overset{\text{def}}{=} \inf_{\substack{\Im\in\Pi \\ \mathrm{sdp}(\Im)\geqslant\alpha}} \mathrm{opt}(\Im)\,.$$

Theorem 1.2. *Let $\Im$ be an instance of the CSP Π. Suppose $\mathrm{sdp}(\Im) \geqslant \alpha$. Then for every $\varepsilon > 0$, we can compute an assignment for $\Im$ of value $\mathrm{gap}(\Pi;\alpha-\varepsilon)-\varepsilon$ in time*

$$m \cdot 2^{2^{\mathrm{poly}(kq/\varepsilon)}} \cdot \mathrm{polylog}\, n\,.$$

Notice that the doubly-exponential factor in the running time depends only on the CSP Π and not on the instance $\Im$. Hence, if we fix the CSP Π, this factor is only a constant factor in the running time (albeit a very large one).

We remark that for specific CSPs, the doubly-exponential factor can sometimes be avoided. Instead of combining Theorem 1.1 with the generic rounding algorithm of [RS09a], we can combine the theorem with other rounding algorithms which are known for specific CSPs. Many of these rounding procedures can be implemented very efficiently (e.g., the hyperplane rounding of [GW95]).

The approximation guarantee of the algorithm in Theorem 1.2 is optimal assuming the Unique Games Conjecture. More precisely, Raghavendra [Rag08] showed that for every $\alpha \in [-1,1]$ and every $\varepsilon > 0$, it is UG-hard to distinguish for an instance $\Im \in \Pi$ between the case $\mathrm{opt}(\Im) \geqslant \alpha - \varepsilon$ and the case $\mathrm{opt}(\Im) \leqslant \mathrm{gap}(\Pi;\alpha)+\varepsilon$. Assuming the UGC, this result implies that given an instance $\Im \in \Pi$ with $\mathrm{opt}(\Im) \geqslant \alpha - \varepsilon$, it is NP-hard to find an assignment for $\Im$ of value $\mathrm{gap}(\Pi,\alpha)+\varepsilon$.

Even without assuming the Unique Games Conjecture, the approximation guarantee of the algorithm in Theorem 1.2 is not worse than the integrality gap of very strong SDP relaxations for CSPs. Raghavendra and Steurer [RS09b] consider a hierarchy of SDP relaxations $\{\mathrm{sdp}_R\}_{R\in\mathbb{N}}$ (the R^{th} relaxation contains at least all valid linear inequalities on up to R variables) and they show that for every $R \in \mathbb{N}$ and every $\varepsilon > 0$, there exists an instance $\Im \in \Pi$ such that $\mathrm{sdp}_R(\Im) \geqslant \alpha - \varepsilon$ but $\mathrm{opt}(\Im) \leqslant \mathrm{gap}(\Pi;\alpha)+\varepsilon$. (Here, R can even be doubly-logarithmic in the size of $\Im$.) In other words, all these strong SDP relaxations have exactly the same integrality gap curve as the basic SDP relaxation that we consider.

1.2 Techniques.

Many of our techniques are borrowed from [AK07] and [RS09a]. In the following, we give a detailed informal description of the algorithm and indicate the points of departure from previous works. The description assumes some familiarity with SDP relaxations and CSPs. (See §2 for basic definitions and notations.)

Solving SDPs via Matrix Multiplicative Weights. Before discussing the specifics of our algorithm for CSPs, let us first consider the following general semidefinite feasibility problem: We are given a system of linear inequalities over the set of *density matrices* (positive semidefinite matrices with trace one) and we desire either an (approximate) solution or a proof that the system is infeasible.

Without loss of generality, this system has the form $X \bullet Y_1, \dots, X \bullet Y_m \geqslant 0$, where X is an unknown density matrix and $Y_1, \dots, Y_m$ are linear constraints. (Note that a linear inequality $A \bullet X \geqslant b$ is equivalent to $(A - bI) \bullet X \geqslant 0$ if X is a density matrix.)

We solve this semidefinite feasibility problem in an iterative manner. Suppose the density matrix X is our current tentative solution. If X satisfies all constraints approximately, say $X \bullet Y_1, \dots, X \bullet Y_m \geqslant -\delta$, we can stop. Otherwise, we can find a constraint such that $X \bullet Y_i < -\delta$. In this case, we update our tentative solution to $X' := f(X, Y_i)$ for some carefully chosen function f, and repeat the process for X'.

The basic goal of the update is to decrease the violation of the constraint corresponding to Y_i. A natural update would be $X' := X + \varepsilon Y_i$ for some parameter $\varepsilon > 0$. (Notice that with this update rule, the algorithm essentially does gradient descent with the goal to maximize the function $F(X) = \min_{1 \leqslant i \leqslant m} X \bullet Y_i$. The only difference is that gradient descent would insist to use the maximally violated constraint for updating the current solution.) One problem with this "linear" update is that the new tentative solution X' might not be a density matrix. Hence, we would have to project $X + \varepsilon Y_i$ onto the set of density matrices, which could be computationally expensive. To avoid these problems, [AK07] choose the update function

$$f(X, Y) := \tfrac{1}{\operatorname{Tr} e^{\log(X)+\varepsilon Y}} e^{\log(X)+\varepsilon Y} .$$

Notice that $f(X, Y)$ is a density matrix for all real symmetric matrices X and Y. Following [AK07], we refer to the iterative algorithm with this update rule as *Matrix Multiplicative Weights* algorithm. Similar or identical update rules have been proposed in several other works, e.g., [DT99, TRW05, WK06]. In fact, this update rule can be derived from general algorithmic frameworks for convex optimization [NY83, BT03]. All works give essentially the same bounds for the convergence of the algorithm:

If the system $X \bullet Y_1, \ldots, X \bullet Y_m \geqslant 0$ has a solution over density matrices, then the Matrix Multiplicative Weights algorithm finds a density matrix X^* with $X^* \bullet Y_1, \ldots, X^* \bullet Y_m \geqslant -\delta$ after at most $O(\rho/\delta)^2 \cdot \log n$ updates (with the right choice of ε). Here, n is the dimension of the matrices and ρ is an upper bound on the spectral norm of the matrices Y_i. The parameter ρ is the *width* of the system $X \bullet Y_1, \ldots, X \bullet Y_m \geqslant 0$.

In applications of the Matrix Multiplicative Weights algorithm, we typically have a set $\mathcal{X}$ of "good" solutions in mind (say all SDP solutions of value at least α for a given CSP instance $\Im$) and this set $\mathcal{X}$ has a natural description as a system of linear inequalities.

However, in most cases, this natural system of linear inequalities has undesirable properties: either a δ-approximate solution to the system is meaningless or the system has very large width.

Hence, the challenge is to find a "good" description of the set $\mathcal{X}$ by linear inequalities. We formalize the notion of a "good" description by the notion of a width-bounded separation oracle: A *ρ-bounded δ-separating oracle* for $\mathcal{X}$ is an algorithm that tries to answer the question "$X \in \mathcal{X}$?" for given a density matrix X. If the oracle outputs **Yes**, then X is a "close" to the set $\mathcal{X}$ (where the precise notion of closeness depends on the specific application). Otherwise, the oracle outputs **No** and returns a ρ-bounded δ-separating hyperplane between X and $\mathcal{X}$, i.e., a matrix Y with $\|Y\| \leqslant \rho$ such that $X \bullet Y < -\delta$ but $X' \bullet Y \geqslant 0$ for all $X' \in \mathcal{X}$.

It is easy to see that a separation oracle for the set $\mathcal{X}$ is sufficient to implement the Matrix Multiplicative Weights algorithm. (To update the current tentative solution X, we only need to know a constraint that is violated by X.)

Good separations for an SDP relaxation of CSPs. The Matrix Multiplicative Weights algorithm allows us to reduce the task of computing an approximately optimal SDP solution for a given CSP instance $\Im$, to the task of finding bounded separating hyperplanes for the set of SDP solutions for $\Im$ with value at least α.

To design a good separation oracle for this set, the first step is to understand when a density matrix X is close to being a good SDP solution. Here, we use techniques from [RS09a].

An SDP solution for a CSP instance $\Im$ consists of collection of vectors and a collection local distributions. For every variable $i \in V$ and every label $a \in \Sigma$, we have a vector $v_{i,a}$ corresponding to the event that variable i is assigned the label a. For every local payoff function $\phi \colon \Sigma^S \to [-1, 1]$, we have a local distribution μ over assignments $x \in \Sigma^S$ to the variable set S. An important type of constraint in the SDP relaxation is that the inner products of the vectors match the degree-2 correlations of the local distributions, i.e., for all $i, j \in S$ and $a, b \in \Sigma$,

$$\langle v_{i,a}, v_{j,b} \rangle = \mathop{\mathbb{E}}_{x \sim \mu} x_i x_j \,. \qquad (1.1)$$

Notice that these constraints are *local*, in the sense that there is no constraint between the vectors of two variables unless they appear in the same local payoff function. The objective in the SDP relaxation is to maximize the value $\mathbb{E}_{x\sim\mu} \phi(x)$ for a typical local payoff function in $\Im$.

A robustness theorem [RS09a] for this SDP relaxation shows that if a (non-feasible) SDP solution satisfies all the constraints (1.1) up to an additive error of ε, then the SDP solution can be modified such it satisfies all constraints exactly, at the cost of decreasing the objective value by at most $\sqrt{\varepsilon} \cdot \operatorname{poly}(kq)$.

The following is an important consequence of this robustness theorem: Consider an SDP solution that satisfies the constraints (1.1) up to a small error *on average*, i.e., the *average violation* of constraints (1.1) is small. Then, (by Markov's inequality,) it has to be the case that for most local payoff functions, all local constraints are satisfied up to a small error. Let us consider a new CSP instance $\Im'$ that contains only those payoff functions whose local constraints are all satisfied up to a small error. By the robustness theorem, the SDP solution with the small average violation can be modified to a feasible SDP solution for this new instance $\Im'$. Since the instances $\Im$ and $\Im'$ differ only in very few local payoff functions, it does not effect the approximation guarantee of our final algorithm if we pass from $\Im$ to $\Im'$.

The discussion so far shows that our separation oracle for CSPs only has to check the average violation of the local constraints (1.1). It turns out that if the average violation of the local constraints is large, then we can efficiently find separating hyperplanes of low width. (The hyperplane will essentially be the average of the violated local constraints.)

In the following, let us say an SDP solution for $\Im$ is *good* if the average violation of the local constraints (1.1) is small and the objective value of the solution is large. When we apply the Matrix Multiplicative Weights algorithm, the natural goal is to find a density matrix X such that $X_{ia,jb} = \langle v_{i,a}, v_{j,b}\rangle / n$ for a collection of vectors $\{v_{i,a}\}$ that can be extended to a good SDP solution for $\Im$. As a consequence of this setup, our separation oracle has to solve the following task: Given a density matrix X, either extend it to a good SDP solution for $\Im$, or find a hyperplane that shows that X cannot be extended to a good SDP solution.

For the purpose of solving this task, we introduce *local linear programs*, one for each local payoff function. The linear program for a local payoff function ϕ tries to find a local distribution μ such that on the one hand μ matches X well (in the sense of (1.1)), and on the other hand the expected payoff of ϕ under μ is as large as possible. These local linear programs have the following nice property: If they fail to extend X to a good SDP solution, then we can combine their optimal dual solutions to a separating hyperplane for X. The fact that the hyperplane is a combination of local dual solutions also allows us to show a good bound on the width of the hyperplane.

One issue that we neglected in the discussion so far is that not all constraints of the SDP relaxation for $\Im$ are local. The missing constraints are captured by the identities $\sum_{a=1}^{q} v_{i,a} = v_0$ for variables $i \in V$. (Here, v_0 is a unit vector.) The proof of the robustness theorem in [RS09a] crucially uses that also these constraints are approximately satisfied. Hence, we need to revise our notion of "good" SDP solutions slightly. In addition to the previous conditions, we require that the average violation of the constraints $\sum_{a=1}^{q} v_{i,a} = v_0$ is small. Another issue is how to integrate the extra vector v_0 into our current setup. We resolve this issue by representing v_0 in an implicit manner, as the (normalized) average of the Gram vectors of X. This implicit representation also allows us to come up with low-width hyperplanes that separate over the constraints involving v_0.

Finding separations rapidly. So far we discussed how to find a good SDP solution for a CSP instance $\Im$ using only a small (logarithmic) number of iterations of the Matrix Multiplicative Weights algorithm. In the following, we outline how to implement each iteration efficiently.

The first step is to approximate the matrix exponential in the update rule by a simpler (low-degree) matrix function. For our applications (as well as for most of the applications in [AK07]), it is straight-forward to argue that the effect of this approximation is negligible.

The second step is to approximate the matrix X as the Gram matrix of low-dimensional vectors. (In this way, we obtain a low-rank approximation $\tilde{X}$ of X.) To obtain the low-dimensional vectors, we project the Gram vectors of X onto a random low-dimensional subspace (Johnson–Lindenstrauss dimensionality reduction [JL84]).

In order to argue that this approximation does not affect the convergence of our algorithm, we need to show that the separation oracle does not distinguish between X and $\tilde{X}$: If the oracle finds a separating hyperplane for $\tilde{X}$, this hyperplane should also separate X.

The separation oracles of Arora and Kale [AK07] only look at pairwise distances of the Gram vectors. Johnson and Lindenstrauss [JL84] showed that random projections preserve pairwise distances arbitrarily well. It follows that the separation oracles of Arora and Kale do not distinguish between X and $\tilde{X}$.

For our CSP separation oracle, we cannot argue in this way, because it is no longer clear that it only depends on the pairwise distances of the vectors. Instead, we bound a certain (simple) norm of the hyperplanes Y that our separation oracle can output. (This norm will depend on the hypergraph structure of the local payoff functions of $\Im$.) Then, we can upper bound $\left|Y \bullet (X - \tilde{X})\right|$ by the dual norm of $X - \tilde{X}$. (The dual norm turns out to be a 1-norm with respect to a certain measure.) Finally, we show that if X is a density matrix, then the Johnson–Lindenstrauss lemma implies that this dual norm of $X - \tilde{X}$ is small.

Irregular instances. In the discussion so far we neglected issues that arise from irregularity of the CSP instance $\Im$, e.g., if some variables appear in significantly more local payoff functions than the typical variable. In this case, the spectral norm of the hyperplanes Y generated by the local linear programs can be very large. However, it turns out that we can bound the spectral norm after weighting the coordinates according to their relative frequency in the local payoff function (the *degree* of the variable). Formally, if D is the diagonal matrix that contains the degrees of the variables, then the spectral norm of $D^{-1/2} Y D^{-1/2}$ is bounded for the hyperplanes Y that our oracle considers. We can adapt the Matrix Multiplicative Weights algorithm to work in this setting. This modified algorithm allows to solve linear inequalities over the set of positive semidefinite matrices X that satisfy $D \bullet X = 1$. The convergence of the algorithm is the same as before with the crucial difference that the width of the separating hyperplanes Y is measured by the degree-weighted spectral norm $\|D^{-1/2} Y D^{-1/2}\|$ (as opposed to the usual spectral norm $\|Y\|$).

2 Preliminaries

Let $\mathcal{M}_n$ denote the set of real symmetric n-by-n matrices. We equip the space $\mathcal{M}_n$ with the inner product $A \bullet B := \operatorname{Tr} AB$ (sometimes called the *Frobenius* or *Hilbert–Schmidt* product). For $A \in \mathcal{M}_n$, we let $\lambda_{\max}(A)$ denote the largest eigenvalue of A. For $A, B \in \mathcal{M}_n$, we write $A \preceq B$ if $A \bullet X \leqslant B \bullet X$ for all positive semidefinite (p.s.d.) $X \in \mathcal{M}_n$. Let $\Delta_n := \{X \in \mathcal{M}_n \mid \operatorname{Tr} X = 1,\ X \succeq 0\}$ denote the set of n-dimensional *density matrices*, a quantum-mechanical analogue of probability distributions. Every matrix $A \in \mathcal{M}_n$ satisfies $\lambda_{\max}(A) = \max_{X \in \Delta_n} A \bullet X$. For a non-negative diagonal matrix $D \in \mathcal{M}_n$, we define a generalization of density matrices, $\Delta_D := \{X \in \mathcal{M}_n \mid D \bullet X,\ X \succeq 0\}$.

2.1 Matrix Multiplicative Weights Bounds.

For $\varepsilon > 0$, let $E_\varepsilon \colon \mathcal{M}_n \to \Delta_n$ be the function

$$E_\varepsilon(Y) \overset{\text{def}}{=} \exp(\varepsilon Y) / \operatorname{Tr} \exp(\varepsilon Y) \,.$$

The Matrix Multiplicative Weights method relies on the following upper bound on the largest eigenvalue of a sum of matrices. We use the notation $Y_{<t} := Y_1 + \ldots + Y_{t-1}$. By the usual convention for empty sums, $Y_{<1} = 0$.

Theorem 2.1. ([AK07]) *Let $\varepsilon > 0$ be small enough and $Y_1, \ldots, Y_T$ be a sequence in $\mathcal{M}_n$ with $0 \preceq Y_t \preceq I$. Then,*

$$\lambda_{\max}(Y_1 + \ldots + Y_T) < (1+\varepsilon) \sum_{t=1}^{T} X_t \bullet Y_t + \tfrac{\log n}{\varepsilon} \,,$$
$$\textit{where } X_t := E_\varepsilon(Y_{<t}) \,.$$

Proof. We estimate the largest eigenvalue by the trace of the exponentiated matrix (a common trick, see for example the proof of the matrix-valued Chernoff bound of Ahlswede and Winter [AW02, WX08]),

$$\begin{aligned}
& e^{\varepsilon \lambda_{\max}(Y_1 + \ldots + Y_T)} \\
& \leqslant \operatorname{Tr} e^{\varepsilon (Y_1 + \ldots + Y_T)} \\
& \leqslant \operatorname{Tr} e^{\varepsilon (Y_1 + \ldots + Y_{T-1})} e^{\varepsilon Y_T} \\
& \leqslant \operatorname{Tr} e^{\varepsilon (Y_1 + \ldots + Y_{T-1})} (I + (e^\varepsilon - 1) Y_T) \\
& = (1 + (e^\varepsilon - 1) X_T \bullet Y_T) \operatorname{Tr} e^{\varepsilon (Y_1 + \ldots + Y_{T-1})} \\
& \leqslant e^{(e^\varepsilon - 1) X_T \bullet Y_T} \operatorname{Tr} e^{\varepsilon (Y_1 + \ldots + Y_{T-1})} \\
& \leqslant \ldots \leqslant e^{(e^\varepsilon - 1) \sum_{t=1}^{T} X_T \bullet Y_T} \cdot n \,.
\end{aligned}$$

The second step uses the Golden–Thompson inequality [Gol65, Tho65]. Since $e^\varepsilon - 1 = \varepsilon + \varepsilon^2/2 + O(\varepsilon^3) < \varepsilon + \varepsilon^2$ for small enough ε, we obtain the desired upper bound on the largest eigenvalue.

Let $D \in \mathcal{M}_n$ be a positive diagonal matrix. (For the first reading, it makes sense to replace all occurrences of D and $D^{-1/2}$ by the identity.)

Corollary 2.2. *Let $\varepsilon > 0$ be small enough and let $Y_1, \ldots, Y_T$ be a sequence in $\mathcal{M}_n$ with $-D \preceq Y_t \preceq D$. Then, for every $X \in \Delta_D$,*

$$\sum_{t=1}^{T} (X - (1+\varepsilon) X_t) \bullet Y_t < \varepsilon T + \tfrac{2 \log n}{\varepsilon} \,,$$
$$\textit{where } X_t := E_{\varepsilon,D}(Y_{<t}) \textit{ and}$$
$$E_{\varepsilon,D}(Y) \overset{\text{def}}{=} D^{-1/2} E_\varepsilon(\tfrac{1}{2} D^{-1/2} Y D^{-1/2}) D^{-1/2} \in \Delta_D \,.$$

Proof. Apply Theorem 2.1 to the matrices

$$Y_t' := \tfrac{1}{2} D^{-1/2} Y_t D^{-1/2} + \tfrac{1}{2} I \,.$$

(Notice that $E_\varepsilon(Y + \alpha I) = E_\varepsilon(Y)$ for all $\alpha \in \mathbb{R}$).

2.2 Semidefinite Feasibility Problems.

Let $\mathcal{X}$ be a convex subset of Δ_D. (We think of $\mathcal{X}$ as the set of feasible solutions to a semidefinite program.)

A *δ-separation oracle* for $\mathcal{X}$ is an algorithm that given a matrix $X \in \Delta_D$ does one of the following things:

Yes the algorithm determines that X is close to the set $\mathcal{X}$. Here, the precise notion of closeness will depend on the particular application.

No the algorithm finds a hyperplane that separates X from the feasible set $\mathcal{X}$ by a δ-margin, i.e., it outputs a matrix $A \in \mathcal{M}_n$ and a scalar $b \in \mathbb{R}$ such that $A \bullet X \leqslant b - \delta$ and on the other hand $A \bullet X' \geqslant b$ for all $X' \in \mathcal{X}$.

A separation oracle is *ρ-bounded* if every A and b in the **No**-case satisfy

$$-\rho D \preceq A - bD \preceq \rho D \,.$$

Note that the parameters ρ and δ are not independent. If we scale the outputs A and b by the factor $1/\rho$, the resulting oracle is 1-bounded and δ/ρ-separating. We find it more convenient to have both parameters.

Basic feasibility algorithm. The Matrix Multiplicative Weights bound suggests an algorithm that either finds a point close to $\mathcal{X}$ or proves that $\mathcal{X}$ is empty. Let ORACLE be a ρ-bounded δ-separation oracle for the set $\mathcal{X}$. Iterate the following for t from 1 to T:

1. Call ORACLE on input $X_t := E_{\varepsilon,D}(Y_{<t})$.
2. If the oracle outputs **Yes**, we stop.
3. Otherwise, the oracle provides a δ-separating hyperplane $A_t \bullet X_t \leqslant b_t - \delta$. In this case, put

$$Y_t := \tfrac{1}{\rho}(A_t - b_t D)\,. \tag{2.2}$$

The following lemma is a direct consequence of Corollary 2.2.

LEMMA 2.3. *Let $\varepsilon \leqslant \delta/2\rho$ and $T \geqslant 2\varepsilon^{-2}\log n$. If $\mathcal{X}$ is non-empty, then* ORACLE *will output* **Yes** *within T iterations of the basic feasibility algorithm.*

Proof. For the sake of contradiction, let $X \in \mathcal{X}$ and suppose ORACLE finds a separating hyperplane for T iterations. By Corollary 2.2,

$$\varepsilon T + \tfrac{2\log n}{\varepsilon} > \sum_{t=1}^{T} (X - (1+\varepsilon)X_t) \bullet Y_t \geqslant \delta/\rho T \geqslant 2\varepsilon T\,,$$

contradicting the premise $T \geqslant 2\varepsilon^{-2}\log n$.

Efficient feasibility algorithm. We modify the basic feasibility algorithm in two ways: First, we approximate the function $E_{\varepsilon,D}$ by a low-degree polynomial. Second, we approximate X_t by a low-rank matrix. See Figure 1 In the figure, d and r are parameters that we will determine later (for our applications, they will be at most logarithmic in the input size), and $P^{\leqslant r}_{\varepsilon,D}(Y)$ is the degree-r approximation of $\exp(\varepsilon/2 D^{-1/2} Y D^{-1/2})$,

$$P^{\leqslant r}_{\varepsilon,D}(Y) \overset{\text{def}}{=} \sum_{i=0}^{r} \tfrac{1}{i!}\left(\tfrac{\varepsilon}{2} D^{-1/2} Y D^{-1/2}\right)^i\,.$$

Before proving the properties of this feasibility algorithm, let us briefly justify the choice of $\tilde{X}_t$. Since the separation oracle expects as input a matrix from Δ_D, the choice of α_t is clear.

The parameters of the random matrix Φ_t are chosen such that $\mathbb{E}\,\Phi_t^{\intercal}\Phi_t = I$. Therefore,

$$\mathbb{E}\,\tilde{W}_t^{\intercal}\tilde{W}_t = D^{-1/2} P^{\leqslant r}_{\varepsilon,D}(1/2 Y_{<t})^2 D^{-1/2}\,.$$

For t from 1 to T, iterate the following:

1. Sample a d-by-n Gaussian matrix Φ_t, with each entry independently chosen from $\mathcal{N}(0, 1/d)$.
2. Compute the d-by-n matrix $\tilde{W}_t$,
$$\tilde{W}_t := \Phi_t \cdot P^{\leqslant r}_{\varepsilon,D}\left(\tfrac{1}{2}Y_{<t}\right) \cdot D^{-1/2}\,.$$
3. Call ORACLE on input $\tilde{X}_t := \alpha_t \tilde{W}_t^{\intercal}\tilde{W}_t \in \Delta_D\,$, where the multiplier $\alpha_t > 0$ is chosen such that $D \bullet \tilde{X}_t = 1$.
4. If ORACLE outputs **Yes**, we stop.
5. Otherwise, the oracle provides a δ-separating hyperplane $A_t \bullet \tilde{X}_t \leqslant b_t - \delta$. Set Y_t as in (2.2).

Figure 1: Efficient feasibility algorithm.

For large enough r, the matrix $P^{\leqslant r}_{\varepsilon,D}(1/2 Y_{<t})^2$ is close to $\exp(\varepsilon/2 D^{-1/2} Y_{<t} D^{-1/2})$. Therefore, the matrix $\mathbb{E}\,\tilde{W}_t^{\intercal}\tilde{W}_t$ is, up to a scaling factor, close to $X_t = E_{\varepsilon,D}(Y_{<t})$.

The following lemma shows that we can compute (a Gram representation of) $\tilde{X}_t$ quickly if matrix-vector multiplication with the matrices A_t is efficient.

LEMMA 2.4. *Suppose we can perform matrix-vector multiplication with the matrices A_t in time $T_{\textit{matrix-vector}}$. Then, for every t, we can compute α_t and $\tilde{W}_t$ in time $O(t \cdot r \cdot d \cdot T_{\textit{matrix-vector}})$.*

Proof. We can compute a row of $\tilde{W}_t$ by multiplying a row of Φ_t to the matrix $P^{\leqslant r}_{\varepsilon,D}(1/2 Y_{<t}) D^{-1/2}$ from the left. Using the structure of $P^{\leqslant r}_{\varepsilon,D}(1/2 Y_{<t})$, this multiplication can be carried out in time $O(t \cdot r \cdot T_{\text{matrix-vector}})$. (Here, we use that matrix multiplication is linear and associative.)

To compute α_t, it is enough to compute the diagonal of $\tilde{W}_t^{\intercal}\tilde{W}_t$, which can be done in time $O(d \cdot n)$ (after computing $\tilde{W}_t$).

We need to prove an analogue of Lemma 2.3 for the efficient feasibility algorithm. It is no longer enough that the oracle is ρ-bounded and δ-separating. In the next lemma, we formulate a simple condition for the oracle that allows us to prove an analogue of Lemma 2.3. (It is cumbersome to check that a concrete separation oracle satisfies this condition. In Definition 2.6, we give a condition that is easier to verify and that implies the condition in the next lemma.)

LEMMA 2.5. *Let $\varepsilon = \delta/4\rho$ and $T = 2\varepsilon^{-2}\log n$. Suppose that for every $t \leqslant T$,*

$$\mathop{\mathbb{P}}_{\Phi_t}\left\{\begin{matrix}\text{ORACLE outputs } \textbf{No} \text{ in iteration } t \\ \text{and } A_t \bullet X_t > b_t - \delta/2\end{matrix}\right\} \ll 1/T\,.$$

Then, if $\mathcal{X}$ is non-empty, ORACLE *will output* **Yes** *with high probability within T iterations of the efficient feasibility algorithm.*

Before proving the lemma, let us first clarify the condition of the lemma. In iteration t, we call ORACLE on matrix $\tilde{X}_t$, which depends on the matrices $Y_{<t}$ and Φ_t. The lemma asserts that for all possibilities of $Y_{<t}$, the probability of the specified event over the randomness of Φ_t is significantly smaller than $1/T$. Notice that if the specified event occurs then the δ-separating hyperplane that ORACLE exhibited for $\tilde{X}_t$ is not even $\delta/2$-separating for X_t. (Here, the matrices X_t are defined as in the basic feasibility algorithm, $X_t = E_{\varepsilon,D}(Y_{<t})$.)

Proof of Lemma 2.5. Suppose $\mathcal{X}$ is non-empty. From the union bound, it follows that with high probability none of the specified events occurs. For the sake of contradiction, assume that ORACLE did not output **Yes** within T iterations. Then, the matrices $Y_1, \ldots, Y_T$ are $\delta/2$-separating hyperplanes for the matrices $X_1, \ldots, X_T$ with $X_t = E_{\varepsilon,D}(Y_{<t})$. Lemma 2.3 shows that these hyperplanes contradict our assumption that $\mathcal{X}$ is non-empty.

Robust Separation Oracle. The following definition abstracts the condition of Lemma 2.5 so that it is easier to verify for a concrete oracle.

DEFINITION 2.6. *We say that a ρ-bounded δ-separation oracle is* d-robust *if for every matrix $X \in \Delta_D$ with $X = W^{\intercal}W$,*

$$\mathop{\mathbb{P}}_{\Phi}\left\{\begin{matrix}\text{ORACLE outputs } \textbf{No} \text{ on input } \tilde{X} \\ \text{and } A \bullet X > b - 3\delta/4\end{matrix}\right\} \ll \tfrac{(\delta/\rho)^2}{\log n}\,.$$

Here, Φ is a d-by-n Gaussian matrix, with each component chosen from $\mathcal{N}(0, 1/d)$, and $\tilde{X} := \alpha W^{\intercal}\Phi^{\intercal}\Phi W$, where multiplier $\alpha > 0$ is chosen such that $D \bullet \tilde{X} = 1$.

LEMMA 2.7. *Let $\varepsilon = \delta/4\rho$ and $T = 2\varepsilon^{-2}\log n$. Suppose we have a d-robust ρ-bounded δ-separation oracle. Then, if we set $r = 10\varepsilon^{-1}\log n$, the oracle satisfies the condition of Lemma 2.5.*

Proof. See §A.2.

The Johnson–Lindenstrauss lemma is used to show that the separation oracle we construct is d-robust for d logarithmic in n.

LEMMA 2.8. ([JL84]) *Let Φ be a d-by-n Gaussian matrix, with each entry independently chosen from $\mathcal{N}(0, 1/d)$. Then, for every vector $u \in \mathbb{R}^n$ and every $\varepsilon \in (0, 1)$,*

$$\mathop{\mathbb{P}}_{\Phi}\left\{\|\Phi u\| = (1 \pm \varepsilon)\|u\|\right\} \geqslant 1 - e^{-\Omega(\varepsilon^2 d)}\,.$$

2.3 Constraint Satisfaction Problems. An instance $\Im$ of a constraint satisfaction problem consists of a set of variables V, an alphabet Σ, and a list of "local" payoff functions $\phi_1, \ldots, \phi_m$ with $\phi_t \colon \Sigma^{S_t} \to [-1, 1]$ for a subset $S_t \subseteq V$. The payoff functions come with non-negative weights $w_1, \ldots, w_m$ such that $\sum_t w_t = 1$. We put $n := |V|$, $q := |\Sigma|$ and $k := \max_t |S_t|$. (We think of the parameters q and k as constant or at least very small compared to n and m.) For an assignment $x \in \Sigma^V$, we let $\Im(x) := \sum_t w_t \phi_t(x)$ denote the average value of a payoff function for the assignment x. (Formally, we extend ϕ_t to Σ^V such that $\phi_t(x) := \phi_t(x_{|S_t})$, where $x_{|S_t}$ is the restriction of the assignment x to the variable set S_t.) Finally, we can define the *(optimal) value* of $\Im$ as $\operatorname{opt}(\Im) := \max_{x \in \Sigma^V} \Im(x)$.

SDP Relaxation. We consider the following semidefinite relaxation $\operatorname{SDP}(\Im)$,

$$\text{(2.3)} \quad \text{maximize} \sum_t w_t \mathop{\mathbb{E}}_{x \sim \mu_t} \phi(x)$$

$$\text{(2.4)} \quad \text{subject to } \langle v_{i,a}, v_{j,b}\rangle = \mathop{\mathbb{P}}_{x \sim \mu_t}\left\{x_i = a, x_j = b\right\}$$

$$(i, j \in S_t,\ \ a, b \in \Sigma,\ \ t \in [m])\,,$$

$$\text{(2.5)} \quad \langle v_{i,a}, v_0\rangle = \mathop{\mathbb{P}}_{x \sim \mu_t}\left\{x_i = a\right\}$$

$$(i \in S_t,\ \ a \in \Sigma,\ \ t \in [m])\,,$$

$$\text{(2.6)} \quad \|v_0\|^2 = 1\,.$$

Here, v_0 is a fixed unit vector and we maximize over all sets of vectors $\{v_{i,a}\}_{i \in V, a \in \Sigma} \subseteq \mathbb{R}^{qn}$ and over all sequences of functions $\mu_1, \ldots, \mu_t \colon \Sigma^{S_t} \to \mathbb{R}$ with $\mu_t \geqslant 0$ and $\sum_{x \in \Sigma^{S_t}} \mu_t(x) = 1$. We can interpret a function μ_t as a distribution over assignments $x \in \Sigma^{S_t}$. The notation $x \sim \mu_t$ means that we sample an assignment $x \in \Sigma^{S_t}$ according to the distribution μ_t. Note that the expressions in (2.3) and (2.4) are indeed linear functions in the values of $\mu_1, \ldots, \mu_m$, because

$$\mathop{\mathbb{E}}_{x \sim \mu_t} \phi(x) = \sum_{x \in \Sigma^{S_t}} \mu_t(x)\phi_t(x)$$

$$\text{and} \quad \mathop{\mathbb{P}}_{x \sim \mu_t}\{x_i = a, x_j = b\} = \sum_{\substack{x \in \Sigma^{S_t} \\ x_i = a, x_j = b}} \mu_t(x)\,.$$

We define $\operatorname{sdp}(\Im)$ to be the optimal objective value of this SDP relaxation.

The vectors of a feasible solution for SDP($\Im$) satisfies the following relation.

Fact 2.9. *For every variable $i \in V$ (that appears in at least one set S_t),*

$$v_0 = \sum_{a\in\Sigma} v_{i,a} .$$

We remark that the constraints (2.5)–(2.6) are implied by the other constraints. In particular, the optimal value sdp($\Im$) would not change if we dropped constraints (2.5)–(2.6).

Robustness of SDP Relaxations. We are interested in the sensitivity of the relaxation sdp($\Im$) to small errors in the constraints (2.4). For $\varepsilon > 0$, consider the semidefinite relaxation $\mathrm{SDP}_\varepsilon(\Im)$,

$$\text{(2.7)} \quad \max \sum_t w_t \mathop{\mathbb{E}}_{x\sim\mu_t} \phi(x)$$

$$\text{(2.8)} \quad \text{s. t.} \left| \langle v_{i,a}, v_{j,b}\rangle - \mathop{\mathbb{P}}_{x\sim\mu_t} \left\{ x_i = a, x_j = b \right\} \right| \leqslant \varepsilon \quad (i,j \in S_t,\ \ a,b \in \Sigma,\ \ t\in[m]) ,$$

$$\text{(2.9)} \quad \left| \langle v_{i,a}, v_0\rangle - \mathop{\mathbb{P}}_{x\sim\mu_t} \left\{ x_i = a \right\} \right| \leqslant \varepsilon \quad (i \in S_t,\ \ a\in\Sigma,\ \ t\in[m]) ,$$

$$\text{(2.10)} \quad \left| \|v_0\|^2 - 1 \right| \leqslant \varepsilon .$$

Let $\mathrm{sdp}_\varepsilon(\Im)$ denote the optimal objective value of this relaxation. The following theorem shows that $\mathrm{sdp}_\varepsilon(\Im)$ is always close to sdp($\Im$).

Theorem 2.10. ([RS09a]) *For any CSP instance $\Im$ with alphabet size k and arity q,*

$$\mathrm{sdp}(\Im) \leqslant \mathrm{sdp}_\varepsilon(\Im) \leqslant \mathrm{sdp}(\Im) + \sqrt{\varepsilon}\cdot \mathrm{poly}(kq) .$$

We remark that the constraints (2.9)–(2.10) seem necessary for this robustness theorem (at least with the current proof in [RS09a]). In contrast, the corresponding exact constraints (2.5)–(2.6) are implied in SDP($\Im$).

3 SDP Algorithm for CSPs

3.1 A Local Linear Program. Let S be a set of k variables and, as before, let Σ be an alphabet of q symbols. Let $\mathcal{M}_{S\times\Sigma}$ denote the set of real symmetric matrices with rows and columns indexed by pairs $(i,a) \in S\times\Sigma$. For a payoff function $\phi\colon \Sigma^S \to [-1,1]$ and a matrix $X\in\mathcal{M}_{S\times\Sigma}$, let $\alpha(X,\phi)$ be the optimal objective value of the following linear program,

$$\text{(3.11)} \quad \text{maximize} \quad \mathop{\mathbb{E}}_{x\sim\mu} \phi(x) - C\cdot\|X - M(\mu)\|_\infty .$$

Here, the maximum is over all distributions μ on Σ^S, C is a parameter that we will determine later and M is the linear mapping from $\Sigma^S\to\mathbb{R}$ to $\mathcal{M}_{S\times\Sigma}$ defined as

$$M(\mu)_{iajb} \stackrel{\text{def}}{=} \sum_{\substack{x\in\Sigma^S\\ x_i=a,\ x_j=b}} \mu(x) .$$

We denote by $\mu(X,\phi)$ an optimal solution to the linear program. Note that if μ is a distribution, then $M(\mu)_{iajb}$ is the probability of the event $\{x_i = a,\ x_j = b\}$ for $x\sim\mu$.

The norm $\|X - M(\mu)\|_\infty$ measures the maximum violation of the constraints (2.4). More precisely, the relaxed constraint (2.8) is equivalent to $\|X-M(\mu)\|_\infty \leqslant \varepsilon$. We remark that the local linear program (3.11) is essentially equivalent to maximizing $\mathbb{E}_{x\sim\mu}\phi(x)$ over all distributions μ that satisfy $\|X - M(\mu)\|_\infty \leqslant 1/C$. One advantage of the formulation (3.11) is that the program is always feasible.

By linear programming duality (see §A.1),

$$\text{(3.12)} \quad \alpha(X,\phi) = \min_{\substack{Y\in\mathcal{M}_{S\times\Sigma}\\ \|Y\|_1\leqslant C}} X\bullet Y + \max_{\Sigma^S}\{\phi - M^*(Y)\} .$$

Here, M^* is the adjoint of M (a linear mapping from $\mathcal{M}_{S\times\Sigma}$ to $\Sigma^S\to\mathbb{R}$),

$$M^*(Y)(x) = \sum_{ij} Y_{ix_ijx_j} .$$

We denote by $Y(X,\phi)$ a minimizer of (3.12).

Properties of local linear programs. Let $\Im$ be a CSP instance as in Section 2.3. For convenience, we assume that $|S_t| = k$ for all $t\in[m]$. Let $\{v_{i,a}\}_{i\in V, a\in\Sigma}$ be a collection of vectors. Let $X_t\in\mathcal{M}_{S_t\times\Sigma}$ denote the Gram matrix of the vectors for the variables in S_t. (Recall that $\Im$ is specified by local payoff functions $\phi_1,\ldots,\phi_m$ and $S_t\subseteq V$ is the set of variables that the local payoff function ϕ_t depends on.)

Fact 3.1. *If the set of vectors $\{v_{i,a}\}_{i\in V,a\in\Sigma}$ can be extended to a feasible solution for* SDP($\Im$) *of value α, then*

$$\sum_{t=1}^m w_t\cdot\alpha(X_t,\phi_t) \geqslant \alpha .$$

The following lemma is a partial converse of the previous fact.

Lemma 3.2. *Suppose $C = 3/\delta$ and*

$$\text{(3.13)} \quad \sum_{t=1}^m w_t\cdot\alpha(X_t,\phi_t) \geqslant \alpha - \delta .$$

Furthermore, assume there exists a vector v_0 such that $\|v_0\|^2 = 1 \pm \delta$ and

$$\sum_t w_t \sum_{i\in S_t} \sum_{a\in\Sigma} \left|\langle v_{i,a}, v_0\rangle - \|v_{i,a}\|^2\right| \leqslant \delta\,. \tag{3.14}$$

Then, there exists a subset $T \subseteq [m]$ of weight $w(T) \geqslant 1 - 2\sqrt{\delta}$ such that $\operatorname{sdp}_{\sqrt{\delta}}(\Im_{|T}) \geqslant \alpha - 2\sqrt{\delta}$ and therefore $\operatorname{sdp}(\Im_{|T}) \geqslant \alpha - \operatorname{poly}(kq)\cdot\delta^{1/4}$.

Proof. We may assume $\delta \leqslant 1$. Let $\mu_t = \mu(X_t, \phi_t)$. Let T_1 be the set of indices t such $\|X_t - M(\mu_t)\|_\infty \geqslant \sqrt{\delta}$, and let T_2 be the set of indices t such that

$$\sum_{i\in S_t} \sum_{a\in\Sigma} \left|\langle v_{i,a}, v_0\rangle - \|v_{i,a}\|^2\right| \geqslant \sqrt{\delta}\,.$$

Let $T := [m]\backslash T_1 \cup T_2$ and consider the CSP instance $\Im_{|T}$. By construction, v_0, $\{\boldsymbol{v}_{i,a}\}$, $\{\mu_t\}_{t\in T}$ forms a feasible solution for the relaxation $\operatorname{SDP}_{\sqrt{\delta}}(\Im_{|T})$. It remains to estimate the weight of T. (Here, the weight of T is defined as $w(T) = \sum_{t\in T} w_t$.) Notice that (3.13) implies that

$$\sum_{t=1}^m w_t \|X_t - M(\mu_t)\|_\infty \leqslant (2+\delta)/C \leqslant \delta\,.$$

Therefore, $w(T_1) \leqslant \sqrt{\delta}$ (Markov's inequality). From (3.14), it follows that $w(T_2) \leqslant \sqrt{\delta}$ (again Markov's inequality). We conclude that $w(T) \geqslant 1 - w(T_1) - w(T_2) \geqslant 1 - 2\sqrt{\delta}$. To finish the proof, we need to lower bound $\operatorname{sdp}_\varepsilon(\Im_{|T})$. The feasible solution for $\operatorname{SDP}_\varepsilon(\Im_{|T})$ given by v_0, $\{\boldsymbol{v}_{i,a}\}$, $\{\mu_t\}_{t\in T}$ has value at least

$$\sum_{t\in T} w_t \cdot \alpha(X_t, \phi_t) \geqslant \alpha - \delta - w(T_1) - w(T_2) \geqslant \alpha - 3\sqrt{\delta}\,.$$

3.2 Separation Oracle. Let $\Im$ be a CSP instance as in Section 2.3. For convenience, we assume that $|S_t| = k$ for all $t \in [m]$.

Let d_i be the normalized *degree* of variable i, that is, $d_i := \sum_{t, S_t \ni i} w_t/k$. (Notice that $\sum_{i\in V} d_i = 1$.)

Consider the vector $\boldsymbol{d} \in \mathbb{R}^{V\times\Sigma}$ with components $\boldsymbol{d}_{i,a} = d_i$. Let $D \in \mathcal{M}_{V\times\Sigma}$ be the diagonal matrix corresponding to $\boldsymbol{d}$. Finally, recall that Δ_D denotes the set of p.s.d. matrices $X \in \mathcal{M}_{V\times\Sigma}$ with $D \bullet X = 1$.

In Figure 2, we describe a separation oracle for the set $\mathcal{X} \subseteq \Delta_D$ that consists of all SDP solutions for $\Im$ with objective value at least α. (In Section 2.3, we said that a SDP solution consist of a collection of vectors $\{v_{i,a}\}_{i\in V, a\in\Sigma}$ and a sequence of distributions $\mu_1, \ldots, \mu_m$ over local assignments. Therefore, it is more precise to say that $\mathcal{X}$ is the set of matrices $X \in \mathcal{M}_{V\times\Sigma}$ such that there exists an SDP solution of value at least α with $X_{ia,jb} = \langle v_{i,a}, v_{j,b}\rangle$.)

CSP-Oracle $(\Im, X, \alpha, \delta)$:

1. Let $\{v_{i,a}\}_{i\in V, a\in\Sigma} \subseteq \mathbb{R}^d$ be Gram vectors for X (assumed to be part of the input).
2. For every $t \in [m]$, compute $X_t \in \mathcal{M}_{S_t\times\Sigma}$, the Gram matrix of the vectors for the variables in S_t.
3. For every $t \in [m]$, solve the local linear program (3.11) for X_t and ϕ_t, and compute $\alpha_t = \alpha(X_t, \phi_t)$ and $Y_t = Y(X_t, \phi_t)$. We choose the parameter C of the local LPs as $3/\delta$.
4. If $\sum_t w_t\alpha_t \leqslant \alpha - \delta$, output **No** and return the hyperplane given by the matrix $A := \sum_t w_t Y_t$ and the scalar $b := \alpha - \sum_t w_t \max_{\Sigma^{S_t}} (\phi_t - M^*(Y_t))$.
5. Compute $v_0 = \sum_i d_i \sum_{a\in\Sigma} v_{i,a}$.
6. If $\left|\|v_0\|^2 - 1\right| \geqslant \delta$, output **No** and return the hyperplane corresponding to the violated constraint. (Here, $b := -s$ with $s = \operatorname{sign}\|v_0\|^2 - 1$ and $A := -s\cdot\boldsymbol{d}\boldsymbol{d}^\intercal$, where $\boldsymbol{d} \in \mathbb{R}^{V\times\Sigma}$ has components $\boldsymbol{d}_{i,a} := d_i$.)
7. If $\sum_i d_i \sum_a \left|\langle v_{i,a}, v_0\rangle - \|v_{i,a}\|^2\right| \geqslant \delta$, output **No** and return the hyperplane corresponding to the violated constraint. (Here, $b := 0$ and $A := -\boldsymbol{d}\cdot\boldsymbol{d}^\intercal \star \boldsymbol{S} - D^{1/2}\operatorname{diag}(\boldsymbol{s})\,D^{1/2}$, where $\boldsymbol{s}_{i,a}$ is the sign of $\langle v_{i,a}, v_0\rangle - \|v_{i,a}\|^2$, $\boldsymbol{S}$ is the matrix $\boldsymbol{S}_{ia,jb} := (\boldsymbol{s}_{i,a} + \boldsymbol{s}_{j,b})/2$, and $\star$ denotes componentwise multiplication.)
8. Otherwise, output **Yes**.

Figure 2: Separation oracle for CSPs.

Lemma 3.3. *The oracle is δ-separating for the set $\mathcal{X}$.*

Proof. In step 6 and step 7, it is clear that we return a constraint that is valid for the set $\mathcal{X}$ (see (2.5)–(2.6) and Fact 2.9). On the other hand, the input matrix X violates the constraint by δ in these cases.

Let us verify that also step 4 is correct. Let $X' \in \mathcal{X}$ be a feasible SDP solution of value at least α. Let $\alpha'_t = \alpha(X'_t, \phi_t)$ where $X'_t \in \mathcal{M}_{S_t\times\Sigma}$ is defined analogous to X_t. Since X' is a feasible and has value at least α, we have $\sum_t w_t\alpha'_t \geqslant \alpha$ (see Fact 3.1). However, from the dual characterization of $\alpha(\cdot,\cdot)$ it follows that $\alpha'_t \leqslant X'_t \bullet Y_t + \max(\phi_t - M^*(Y_t))$. Thus, $\sum_t w_t X'_t \bullet Y_t \geqslant \alpha - \sum_t w_t \max(\phi_t - M^*(Y_t))$. Hence, the constraint in step 4 is valid for the set $\mathcal{X}$. On the other hand, if $\sum_t w_t\alpha_t \leqslant \alpha - \delta$, the input matrix X violates this constraint by at least δ.

Lemma 3.4. *If the oracle outputs* **Yes**, *then there exists a set $T \subseteq [m]$ with $w(T) \geqslant 1 - 2\sqrt{\delta}$ such that the restriction $\Im_{|T}$ of the CSP instance $\Im$ to the payoff functions $\{\phi_t \mid t \in T\}$ satisfies*

$$\mathrm{sdp}_{\sqrt{\delta}}(\Im_{|T}) \geqslant \alpha - \mathrm{poly}(kq)\sqrt{\delta}$$

and therefore $\mathrm{sdp}(\Im_{|T}) \geqslant \alpha - \delta^{1/4} \cdot \mathrm{poly}(kq)$ (using Theorem 2.10). Furthermore, given the Gram vectors $\{v_{i,a}\}$ of the input matrix X for the oracle, we can compute in time $m \cdot d \cdot \mathrm{poly}(q^k)$ a feasible solution for $\mathrm{SDP}(\Im_{|T})$ of value at least $\alpha - \delta^{1/4} \cdot \mathrm{poly}(kq)$.

Proof. Immediate from Lemma 3.2 and the fact that the proof of the robustness theorem in [RS09a] yields an efficient algorithm that converts a solution for $\mathrm{SDP}_{\varepsilon}(\Im_{|T})$ to a solution for $\mathrm{SDP}(\Im_{|T})$.

Lemma 3.5. *The oracle is ρ-bounded for $\rho = \mathrm{poly}(kq/\delta)$.*

Proof. Let us consider the matrix $A = \sum_t w_t Y_t$. Since $\|Y_t\|_1 \leqslant C$, the spectral norm of Y_t is also bounded by C. Let I_t be the identity matrix in $\mathcal{M}_{V\times\Sigma}$ restricted to the rows corresponding to the variables in S_t. Let X' be any p.s.d. matrix in $\mathcal{M}_{V\times\Sigma}$. We have to show that $|A \bullet X'| \leqslant \rho D \bullet X'$. Since the spectral norm of Y_t is bounded by C, we have $|A \bullet X'| \leqslant \sum_t w_t C \cdot I_t \bullet X' = Ck \cdot D \bullet X'$, as desired. For the remaining cases, the calculations are similar (see end of §A.3 for details).

Lemma 3.6. *The algorithm* CSP-Oracle *can be implemented to run in time*

$$m \cdot d \cdot \mathrm{poly}(q^k)\,,$$

assuming the input matrix X is represented by d-dimensional Gram vectors. If the algorithms outputs **No** *it returns (an implicit representation of) a matrix for which we can compute matrix-vector products in time $m \cdot \mathrm{poly}(kq)$.*

Proof. The most expensive part of the algorithm is to solve the local linear programs. Using a polynomial time algorithm for solving linear programs, each local LP can be solved in time $q^{O(k)}$. All computations involving the vectors (computing the local Gram matrices X_t, computing the vector v_0, ...) can be carried out in time $m \cdot d \cdot \mathrm{poly}(kq)$. The matrix $\sum_t w_t Y_t$ has at most mkq non-zero entries. Hence, matrix-vector multiplication is easy. The matrices corresponding to the expressions $\sum_{ij} d_i d_j \sum_{ab}\langle v_{i,a}, v_{j,b}\rangle$ or $\sum_{i\in S}\sum_j d_i d_j \sum_{ab} s_{i,a}\langle v_{i,a}, v_{j,b}\rangle$ are not sparse, but they have low-rank. Again, matrix-vector multiplication is easy.

3.3 Robustness of the Separation Oracle.

In the following, we will show that CSP-Oracle is d-robust for $d = \mathrm{poly}(kq/\delta) \cdot \log n$.

Let $\mathcal{Y}$ be the set of matrices $Y = A - bD$ that CSP-Oracle could possibly output as separating hyperplanes. Let us define a norm on $\mathcal{M}_{V\times\Sigma}$ by

$$\|Z\|_{\mathcal{Y}} \overset{\mathrm{def}}{=} \sup_{Y\in\mathcal{Y}} |Y \bullet Z|\,.$$

Let $X \in \Delta_D$ with $X = W^{\intercal}W$. Let $\{w_{i,a}\}_{i\in V, a\in\Sigma}$ be the columns of W and let S be the set of variables with $\|w_{i,a}\|^2 \geqslant 4$ for some $a \in \Sigma$. Let $\tilde{X} := \alpha W \Phi^{\intercal}\Phi W$, where $\alpha > 0$ is chosen such that $D \bullet \tilde{X} = 1$ and Φ is a Gaussian matrix with d rows and columns indexed by $V \times \Sigma$. The components of Φ are chosen independently from $\mathcal{N}(0, 1/d)$. Let $\{\tilde{w}_{i,a}\}_{i\in V, a\in\Sigma}$ be the columns of $\tilde{W} := \sqrt{\alpha}\,\Phi W$.

We show that CSP-Oracle is robust by proving that $\|X - \tilde{X}\|_{\mathcal{Y}}$ is at most $\delta/4$ with high probability, say $1 - 1/n$.

The first step of the proof is to upper bound $\|\cdot\|_{\mathcal{Y}}$ by a simpler norm. The reader can verify the following lemma in a straight-forward way, by considering each of the three types of hyperplanes that CSP-Oracle can output (corresponding to step 4, step 6, or step 7). For the sake of completeness, we present a proof in Appendix A.3.

Lemma 3.7.

$$\|Z\|_{\mathcal{Y}} \leqslant \mathrm{poly}(kq/\delta) \cdot \Big(\sum_t w_t \sum_{i,j\in S_t}\sum_{a,b}|z_{ia,jb}| + \sum_{ij} d_i d_j \sum_{ab}|z_{ia,jb}| + \sum_i d_i \sum_a |z_{ia,ia}|\Big)\,. \tag{3.15}$$

Lemma 3.8. *For some $d = \mathrm{poly}(kq/\delta) \cdot \log n$,*

$$\mathop{\mathbb{P}}_{\Phi}\left\{\|X - \tilde{X}\|_{\mathcal{Y}} \leqslant \tfrac{\delta}{4}\right\} \geqslant 1 - \tfrac{1}{n}.$$

Proof. Let $\gamma > 0$ (a parameter we determine later). By Lemma 2.8, with probability $p_{\mathrm{JL}} = 1 - O(qn)^2 \cdot e^{-\Omega(\gamma^2 d)}$, all $i, j \in V$ and $a, b \in \Sigma$ satisfy

$$\|\tilde{w}_{i,a} + \tilde{w}_{j,b}\|^2 = (1\pm\gamma)\|w_{i,a} + w_{j,b}\|^2$$
$$\|\tilde{w}_{i,a} - \tilde{w}_{j,b}\|^2 = (1\pm\gamma)\|w_{i,a} - w_{j,b}\|^2$$

In this case, we can bound $Z = X - \tilde{X}$ as follows

$$\begin{aligned}|z_{ia,jb}| &= \tfrac{1}{4}\Big|\|w_{i,a} + w_{j,b}\|^2 - \|w_{i,a} - w_{j,b}\|^2 \\ &\qquad - \|\tilde{w}_{i,a} + \tilde{w}_{j,b}\|^2 + \|\tilde{w}_{i,a} - \tilde{w}_{j,b}\|^2\Big| \\ &\leqslant \tfrac{1}{4}\gamma \cdot \left(\|w_{i,a} + w_{j,b}\|^2 + \|w_{i,a} - w_{j,b}\|^2\right) \\ &= \tfrac{1}{2}\gamma \cdot \left(\|w_{i,a}\|^2 + \|w_{j,b}\|^2\right)\end{aligned}$$

Combining this bound of $|z_{ia,jb}|$ with Lemma 3.7, yields

$$\|Z\|_{\mathcal{Y}} \leqslant \gamma \cdot \mathrm{poly}(kq/\delta) \cdot \sum_i d_i \sum_a \|w_{i,a}\|^2 .$$

(Here, we use that the column and row marginal distributions of the measures in the bound of Lemma 3.7 are equal to the distribution $(d_1, \ldots, d_n)$.)

Since $D \bullet X = 1$, we can choose $\gamma = \mathrm{poly}(\delta/kq)$ such that $\|Z\|_{\mathcal{Y}} \leqslant \delta/4$. Thus, for some $d = O(\log qn/\gamma^2) = \mathrm{poly}(kq/\delta) \cdot \log n$, the success probability is $p_{\mathrm{JL}} \geqslant 1 - 1/n$ as desired.

Lemma 3.9. *For every $\delta > 0$ and for some $d = \mathrm{poly}(qk/\delta) \cdot \log n$ and $\rho = \mathrm{poly}(qk/\delta)$, CSP-Oracle is a d-robust ρ-bounded δ-separation oracle for the set $\mathcal{X}$ (the set of SDP solutions for $\Im$ of value at least α).*

Proof. Lemma 3.3 and Lemma 3.5 show that CSP-Oracle is ρ-bounded and δ-separating. To verify that CSP-Oracle is d-robust we need to upper bound the probability of the event,

$$\mathcal{E} := \left\{ \begin{array}{c} \text{Oracle outputs } \textbf{No} \text{ on input } \tilde{X} \\ \text{and } A \bullet X > b - 3\delta/4\delta \end{array} \right\} .$$

Whenever $\mathcal{E}$ occurs, we have $(A - bD) \bullet \tilde{X} \leqslant -\delta$ but $(A - bD) \bullet \tilde{X} > -3\delta/4$. Hence, the event $\mathcal{E}$ implies that $\|X - \tilde{X}\|_{\mathcal{Y}} > \delta/4$. Using Lemma 3.8, we can conclude that $\mathbb{P}\{\mathcal{E}\} \leqslant 1/n$, as desired.

3.4 Putting things together.

Proof of Theorem 1.1. Let $\Im$ be a CSP instance as in Section 2.3. Suppose we know a good estimate $\alpha \in [-1, 1]$ of $\mathrm{sdp}(\Im)$, say $\alpha \leqslant \mathrm{sdp}(\Im) \leqslant \alpha + \varepsilon$. (If we do not have a good estimate, we can find one by searching through $2/\varepsilon$ possibilities for α.) We apply the efficient feasibility algorithm in Section 2.2 (based on Matrix Multiplicative Weights) with CSP-Oracle as separation oracle. Lemma 3.9 shows that CSP-Oracle is d-robust ρ-bounded and δ-separating (for every $\delta > 0$ and appropriate d and ρ). From Lemma 2.7 and Lemma 2.5, it follows that CSP-Oracle outputs **Yes** after $O(\rho/\delta)^2 \cdot \log n$ iterations of the efficient feasibility algorithm. Lemma 3.4 shows that we can convert the final matrix (that the oracle accepted) to a feasible solution for $\mathrm{SDP}(\Im_{|T})$ with value at least $\alpha - \delta^{1/4}\mathrm{poly}(kq)$. Here, T is a subset of the local payoff functions of $\Im$ of weight at least $1 - 2\sqrt{\delta}$. Let us choose $\delta = \varepsilon^4/\mathrm{poly}(kq)$ such that the value of this SDP solution is at least $\alpha - \varepsilon$ and the weight of T is at least $1 - \varepsilon$. Combining Lemma 3.6 and Lemma 2.4, we see that each iteration of the efficient feasibility algorithm runs in time $m \cdot \mathrm{poly}(q^k/\delta) \cdot \mathrm{polylog}\, n$ (if we choose the parameters r and d appropriately). It follows that the overall running time is bounded by $m \cdot \mathrm{poly}(q^k/\varepsilon) \cdot \mathrm{polylog}\, n$.

Proof of Theorem 1.2. Let $\Im$ be a CSP instance as before and suppose that $\mathrm{sdp}(\Im) \geqslant \alpha$. By Theorem 1.1, we can compute a feasible SDP solution of value $\alpha - \varepsilon$ for an instance $\Im_{|T}$, where T is a subset of the local payoff functions of $\Im$ of weight at least $1 - \varepsilon$. By [RS09a, Theorem 1.1], we can round this SDP solution to an assignment of value $\mathrm{gap}(\Pi; \alpha - 2\varepsilon) - \varepsilon$ for $\Im_{|T}$ in time $m \cdot \exp\exp\mathrm{poly}(kq/\varepsilon) \cdot \mathrm{polylog}\, n$. Since T has weight $1 - \varepsilon$, this assignment has value at least $\mathrm{gap}(\Pi; \alpha - 2\varepsilon) - 3\varepsilon$ for $\Im$.

Acknowledgments. Many thanks to Sanjeev Arora, Satyen Kale, Prasad Raghavendra, and Nisheeth Vishnoi for invaluable discussions about this topic.

References

[AK07] Sanjeev Arora and Satyen Kale, *A combinatorial, primal-dual approach to semidefinite programs*, STOC, 2007, pp. 227–236.

[AW02] Rudolf Ahlswede and Andreas Winter, *Strong converse for identification via quantum channels*, IEEE Transactions on Information Theory **48** (2002), no. 3, 569–579.

[BT03] Amir Beck and Marc Teboulle, *Mirror descent and nonlinear projected subgradient methods for convex optimization*, Oper. Res. Lett. **31** (2003), no. 3, 167–175.

[CMM06a] Moses Charikar, Konstantin Makarychev, and Yury Makarychev, *Near-optimal algorithms for unique games*, STOC, 2006, pp. 205–214.

[CMM06b] Eden Chlamtac, Konstantin Makarychev, and Yury Makarychev, *How to play unique games using embeddings*, FOCS, 2006, pp. 687–696.

[CMM07] Moses Charikar, Konstantin Makarychev, and Yury Makarychev, *Near-optimal algorithms for maximum constraint satisfaction problems*, SODA, 2007, pp. 62–68.

[CW04] Moses Charikar and Anthony Wirth, *Maximizing quadratic programs: Extending grothendieck's inequality*, FOCS, 2004, pp. 54–60.

[DT99] Moshe Doljansky and Marc Teboulle, *An interior proximal algorithm and the exponential multiplier method for semidefinite programming*, SIAM J. Optim. **9** (1999), no. 1, 1–13 (electronic).

[FG95] Uriel Feige and Michel X. Goemans, *Aproximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT*, ISTCS, 1995, pp. 182–189.

[FJ97] A. Frieze and M. Jerrum, *Improved approximation algorithms for MAX k-CUT and MAX BISECTION*, Algorithmica **18** (1997), no. 1, 67–81.

[Gol65] Sidney Golden, *Lower bounds for the Helmholtz function*, Phys. Rev. **137** (1965), no. 4B, B1127–B1128.

[GW95] Michel X. Goemans and David P. Williamson, *Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming*, J. ACM **42** (1995), no. 6, 1115–1145.

[GW04] Michel X. Goemans and David P. Williamson, *Approximation algorithms for MAX-3-CUT and other problems via complex semidefinite programming*, J. Comput. System Sci. **68** (2004), no. 2, 442–470.

[Has05] Gustav Hast, *Beating a random assignment*, APPROX-RANDOM, 2005, pp. 134–145.

[HZ01] Eran Halperin and Uri Zwick, *Approximation algorithms for MAX 4-SAT and rounding procedures for semidefinite programs*, J. Algorithms **40** (2001), no. 2, 184–211.

[JL84] William B. Johnson and Joram Lindenstrauss, *Extensions of Lipschitz mappings into a Hilbert space*, Conference in modern analysis and probability (New Haven, Conn., 1982), Contemp. Math., vol. 26, Amer. Math. Soc., Providence, RI, 1984, pp. 189–206.

[Kho02] Subhash Khot, *On the power of unique 2-prover 1-round games*, STOC, 2002, pp. 767–775.

[KS09] Subhash Khot and Rishi Saket, *SDP integrality gaps with local l1-embeddability*, FOCS, 2009, To appear.

[KZ97] Howard J. Karloff and Uri Zwick, *A 7/8-approximation algorithm for max 3sat?*, FOCS, 1997, pp. 406–415.

[LLZ02] Michael Lewin, Dror Livnat, and Uri Zwick, *Improved rounding techniques for the max 2-sat and max di-cut problems*, IPCO, 2002, pp. 67–82.

[MM01] Shiro Matuura and Tomomi Matsui, *0.863-approximation algorithm for MAX DICUT*, RANDOM-APPROX, 2001, pp. 138–146.

[NY83] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization*, A Wiley-Interscience Publication, John Wiley & Sons Inc., New York, 1983, Translated from the Russian and with a preface by E. R. Dawson, Wiley-Interscience Series in Discrete Mathematics.

[Rag08] Prasad Raghavendra, *Optimal algorithms and inapproximability results for every CSP?*, STOC, 2008, pp. 245–254.

[RS09a] Prasad Raghavendra and David Steurer, *How to round any CSP*, FOCS, 2009, To appear.

[RS09b] ———, *Integrality gaps for strong SDP relaxations of unique games*, FOCS, 2009, To appear.

[Tho65] Colin J. Thompson, *Inequality with applications in statistical mechanics*, Journal of Mathematical Physics **6** (1965), no. 11, 1812–1813.

[Tre09] Luca Trevisan, *Max Cut and the smallest eigenvalue*, STOC, 2009, pp. 263–272.

[TRW05] Koji Tsuda, Gunnar Rätsch, and Manfred K. Warmuth, *Matrix exponentiated gradient updates for on-line learning and bregman projection*, Journal of Machine Learning Research **6** (2005), 995–1018.

[TSSW00] Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson, *Gadgets, approximation, and linear programming*, SIAM J. Comput. **29** (2000), no. 6, 2074–2097 (electronic).

[WK06] Manfred K. Warmuth and Dima Kuzmin, *Online variance minimization*, COLT, 2006, pp. 514–528.

[WX08] Avi Wigderson and David Xiao, *Derandomizing the ahlswede-winter matrix-valued chernoff bound using pessimistic estimators, and applications*, Theory of Computing **4** (2008), no. 1, 53–76.

[Zwi98a] Uri Zwick, *Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint*, SODA, 1998, pp. 201–210.

[Zwi98b] ———, *Finding almost-satisfying assignments*, STOC, 1998, pp. 551–560.

[Zwi99] ———, *Outward rotations: A tool for rounding solutions of semidefinite programming relaxations, with applications to max cut and other problems*, STOC, 1999, pp. 679–687.

A Appendix

A.1 Dual of the Local Linear Program

$$\begin{aligned}&\max_{\mu}\langle \mu,\phi\rangle - C\|X-M(\mu)\|_\infty\\ &=\max_{\mu}\min_{Y}\langle \mu,\phi\rangle + C\langle X-M(\mu),Y\rangle\\ &=\min_{Y}\max_{\mu}\langle \mu,\phi\rangle + C\langle X-M(\mu),Y\rangle\\ &=\min_{Y}\langle X, C\cdot Y\rangle + \max_{\mu}\langle \mu,\phi\rangle - \langle \mu, M^*(C\cdot Y)\rangle\\ &=\min_{Y}\langle X, C\cdot Y\rangle + \max_{\Sigma^S}\{\phi - M^*(C\cdot Y)\}\,.\end{aligned}$$

Here, μ ranges over all distributions on Σ^S and Y ranges over all matrices in $\mathcal{M}_{S\times\Sigma}$ with $\|Y\|_1 = 1$. All inner products refer to the natural coordinate-wise inner products. In the second equation, we use linear programming duality. In the third equation, we use that $\langle M(\mu),Y\rangle = \langle \mu, M^*(Y)\rangle$ for all μ and Y. (This identity is the same as the familiar fact $(Ax)^\intercal y = x^\intercal(A^\intercal y)$.)

A.2 Robust Separation Oracle. Let $P_\varepsilon^{\leqslant r}(Y)$ be the degree-r approximation of $e^{\varepsilon Y}$,

$$P_\varepsilon^{\leqslant r}(Y) \overset{\text{def}}{=} \sum_{i=0}^{r} \tfrac{1}{i!}\,(\varepsilon Y)^i\ .$$

Lemma A.1. (see e.g. [AK07]) *For every $Y\in\mathcal{M}_n$,*

$$\|e^{\varepsilon Y} - P_\varepsilon^{\leqslant r}(Y)\| \leqslant e^{\varepsilon\|Y\|}\cdot e^{-r}\,.$$

Lemma A.2. *Let $\varepsilon = \delta/4\rho$ and $T = 2\varepsilon^{-2}\log n$. Let $r \geqslant 10\varepsilon^{-1}\log n$. For every $t\leqslant T$, define matrices $W_t', X_t' \in \mathcal{M}_n$,*

$$\begin{aligned}W_t' &:= P_{\varepsilon,D}(\tfrac12 Y_{<t})D^{-1/2}\\ X_t' &:= \alpha_t' W_t'^{\intercal} W_t'\,,\end{aligned}$$

where the multiplier $\alpha'_t > 0$ is chosen such that $X'_t \in \Delta_D$. Then, for every $Y' \in \mathcal{M}_n$ with $-\rho D \preceq Y' \preceq \rho D$,

$$|Y' \bullet (X_t - X'_t)| \leqslant \tfrac{1}{n}\,.$$

Proof. Fix a particular $t \leqslant T$. The only property of the matrix $Y_{<t}$ that we use is $-T \cdot D \preceq Y_{<t} \preceq T \cdot D$. Let $Y := D^{-1/2} Y_{<t} D^{-1/2}$ and $Z := P_\varepsilon^{\leqslant r}\left({}^1\!/_2 Y\right)^2 - e^{\varepsilon Y}$. Using Lemma A.1, we bound the spectral norm of Z,

$$\begin{aligned}\|Z\| &= \Big\| \left(P_\varepsilon^{\leqslant r}\left(\tfrac12 Y\right) - e^{\varepsilon Y/2}\right) P_\varepsilon^{\leqslant r}\left(\tfrac12 Y\right) \\ &\quad + e^{\varepsilon Y/2}\left(P_\varepsilon^{\leqslant r}\left(\tfrac12 Y\right) - e^{\varepsilon Y/2}\right)\Big\| \\ &\leqslant 2 \cdot \|P_\varepsilon^{\leqslant r}\left(\tfrac12 Y\right) - e^{\varepsilon Y/2}\| \cdot \|e^{\varepsilon Y/2}\| \\ &\leqslant 2e^{\varepsilon\|Y\|} e^{-r} \leqslant 2e^{\varepsilon T - r}\,.\end{aligned}$$

Next, let us estimate the difference of the normalization factors $\alpha := 1/\operatorname{Tr} e^{\varepsilon Y}$ and $\alpha' := \alpha'_t = 1/\operatorname{Tr} P_\varepsilon^{\leqslant r}\left({}^1\!/_2 Y\right)^2$.

$$|1/\alpha - 1/\alpha'| = |\operatorname{Tr} Z| \leqslant n\|Z\| \leqslant 2ne^{\varepsilon T - r}\,.$$

Since $0 \leqslant \alpha \leqslant {}^1\!/_n$, it follows that $|\alpha - \alpha'| \leqslant 2e^{\varepsilon T - r}$. Therefore, $\|D^{1/2}(X'_t - X_t)D^{1/2}\| \leqslant \min\{\alpha, \alpha'\}\|Z\| + |\alpha - \alpha'|\|e^{\varepsilon Y}\| \leqslant 4e^{2\varepsilon T - r}$.

We bound $|Y' \bullet (X_t - X'_t)|$ as follows

$$\begin{aligned}|Y \bullet (X_t - \tilde{X}'_t)| &\leqslant \|D^{-1/2} Y D^{-1/2}\| \\ &\qquad \cdot \|D^{1/2}(X_t - \tilde{X}'_t)D^{1/2}\|_* \\ &\leqslant \rho \cdot n \cdot \|D^{1/2}(X_t - \tilde{X}'_t)D^{1/2}\| \\ &\leqslant \rho n 4e^{2\varepsilon T - r} \ll 1/n\,.\end{aligned}$$

Here $\|\cdot\|_*$ is the dual norm of the spectral norm (called nuclear norm). It is well-known that this dual norm is equal to the sum of the singular values. Therefore, we can upper bound it by n times the spectral norm.

Lemma A.3. (Restatement of Lemma 2.7) *Let $\varepsilon = \delta/4\rho$ and $T = 2\varepsilon^{-2}\log n$. Suppose we have a d-robust ρ-bounded δ-separation oracle. Then, if we set $r = 10\varepsilon^{-1}\log n$, the oracle satisfies the condition of Lemma 2.5.*

Proof. Since the oracle is d-robust ρ-bounded δ-separating, we have

$$\mathop{\mathbb{P}}_{\Phi}\left\{\begin{matrix}\text{ORACLE outputs } \textbf{No} \text{ on input } \tilde{X}_t \\ \text{and } A \bullet X'_t > b - 3\delta/4\end{matrix}\right\} \ll \frac{(\delta/\rho)^2}{\log n}\,.$$

Recall the notation from Lemma A.2, $W'_t = P_{\varepsilon,D}({}^1\!/_2 Y_{<t})D^{-1/2}$ and $X'_t = \alpha'_t W_t'^{\mathsf{T}} W'_t$. Notice that with this notation, we have $X_t = \alpha_t W_t'^{\mathsf{T}} \Phi_t^{\mathsf{T}} \Phi_t W'_t$.

By Lemma A.2,

$$|(A - bD) \bullet (X'_t - X_t)| \leqslant {}^1\!/_n \leqslant {}^\delta\!/_4\,.$$

Therefore, the event

$$\mathcal{E} := \left\{\begin{matrix}\text{ORACLE outputs } \textbf{No} \text{ on input } \tilde{X}_t \\ \text{and } A \bullet X_t > b - \delta/2\end{matrix}\right\}$$

is a subset of the event

$$\mathcal{E}' := \left\{\begin{matrix}\text{ORACLE outputs } \textbf{No} \text{ on input } \tilde{X}_t \\ \text{and } A \bullet X'_t > b - 3\delta/4\end{matrix}\right\}$$

Thus, $\mathbb{P}\{\mathcal{E}\} \leqslant \mathbb{P}\{\mathcal{E}'\} \ll 1/T$ as desired.

A.3 Robustness of CSP-Oracle.

To prove Lemma 3.7, we first upper bound a certain simple norm of the matrices in $\mathcal{Y}$. Then, we relate the dual of this norm to the right-hand side of Lemma 3.7.

Let μ be the probability measure on $(V \times \Sigma)^2$,

$$\mu_{ia,jb} \overset{\text{def}}{=} \tfrac13 \cdot \tfrac{d_i}{q}\mathbb{I}_{\{i=j,\ a=b\}} + \tfrac13 \cdot \tfrac{d_i d_j}{q^2} + \tfrac13 \cdot \sum_{t:\ S_t \ni i,j} \tfrac{w_t}{(kq)^2}\,.$$

(With probability ${}^1\!/_3$, we pick a diagonal entry according to the degree distribution. With probability ${}^1\!/_3$, we pick a row and a column independently according to the degree distribution. With the remaining probability, we pick an index t with probability w_t and pick two random variables $i, j \in S_t$ and two random labels $a, b \in \Sigma$.)

We will think of μ as a measure on entries of the matrices in $\mathcal{M}_{V\times\Sigma}$.

The next lemma shows that the ∞-norm of the matrices in $\mathcal{Y}$ normalized by the measure μ is bounded.

Lemma A.4. *For every matrix $Y \in \mathcal{Y}$,*

$$\forall i, j \in V.\ \forall a, b \in \Sigma. \quad |Y_{ia,jb}| \leqslant \kappa \cdot \mu_{ia,jb}$$

and $\kappa = \operatorname{poly}(kq/\delta)$.

Before proving this lemma, let us first observe that it has the desired implication.

Proof of Lemma 3.7. We calculate

$$\begin{aligned}\|Z\|_{\mathcal{Y}} &= \sup_{Y\in\mathcal{Y}} |Y \bullet Z| \\ &\leqslant \kappa \sum_{i,j}\sum_{a,b} \mu_{ia,jb}|Z_{ia,jb}| \\ &= \tfrac{\kappa}{3} \cdot \Big(\sum_t w_t \cdot \tfrac{1}{(kq)^2} \sum_{i,j\in S_t}\sum_{a,b} |z_{ia,jb}| \\ &\qquad + \sum_{ij} d_i d_j \cdot \tfrac{1}{q^2} \sum_{ab} |z_{ia,jb}| \\ &\qquad + \sum_i d_i \cdot \tfrac1q \sum_a |z_{ia,ia}|\Big)\,.\end{aligned}$$

The last bound implies the statement of Lemma 3.7 since $\kappa = \operatorname{poly}(kq/\delta)$.

Proof of Lemma A.4. It is easy to check that we can write every matrix $Y \in \mathcal{Y}$ as a sum $Y = \lambda_1 Y^{(1)} + \lambda_2 Y^{(2)} + \lambda_3 Y^{(3)}$ such that the multipliers satisfy $\lambda_1, \lambda_2, \lambda_3 = \mathrm{poly}(kq/\delta)$ and

- the matrix $Y^{(1)}$ has the form $Y^{(1)} = \sum_t w_t Y_t$, where $Y_t \in \mathcal{M}_{S_t \times \Sigma}$ and $\|Y_t\|_1 \leqslant C$,
- the matrix $Y^{(2)}$ has the form $\boldsymbol{d} \cdot \boldsymbol{d}^\intercal \star \boldsymbol{S}$ (for the notation, see CSP-ORACLE in §3.2),
- the matrix $Y^{(3)}$ has the form $D^{-1/2} \mathrm{diag}(\boldsymbol{s}) D^{-1/2}$.

Let us verify that each of these three matrices $Y^{(1)}, Y^{(2)}, Y^{(3)}$ satisfy the bound of the lemma.

Since $\|Y_t\|_1 \leqslant C$ for every $t \in [m]$, we have

$$\left|Y^{(1)}_{ia,jb}\right| \leqslant \sum_{t:\ S_t \ni i,j} w_t \|Y_t\|_\infty \leqslant C \sum_{t:\ S_t \ni i,j} w_t = O(kq/\delta)^2 \mu_{ia,jb}\,.$$

Since $\boldsymbol{S}$ is a sign matrix, we have

$$\left|Y^{(2)}_{ia,jb}\right| \leqslant d_i d_j = O(q^2) \mu_{ia,jb}\,.$$

Since $\boldsymbol{s}$ is a sign vector and $Y^{(3)}$ is diagonal,

$$\left|Y^{(3)}_{ia,jb}\right| \leqslant d_i \mathbb{I}_{\{i=j,\ a=b\}} = O(q) \mu_{ia,jb}\,.$$

We can conclude that the statement of the lemma holds for the every matrix $Y \in \mathcal{Y}$.

We observe that Lemma A.4 and Lemma 3.7 easily imply Lemma 3.7.

Proof of Lemma 3.7. Let $X \in \Delta_D$ and $Y \in \mathcal{Y}$. Then,

$$\begin{aligned} X \bullet Y &\leqslant \kappa \sum_{i,j} \sum_{a,b} \mu_{ia,jb} |X_{ia,jb}| \quad \text{(using Lemma A.4)} \\ &\leqslant \kappa \sum_{i,j} \sum_{a,b} \mu_{ia,jb} \tfrac{1}{2} \left(|X_{ia,ia}| + |X_{jb,jb}|\right) \\ &= \kappa \sum_i d_i \cdot \tfrac{1}{q} \sum_a X_{ia,ia} = \kappa/q \cdot D \bullet X = \kappa/q\,. \end{aligned}$$

We can conclude that $\rho \leqslant \kappa/q = \mathrm{poly}(kq/\delta)$, because

$$\rho = \sup_{Y \in \mathcal{Y}} \sup_{X \in \Delta_D} |X \bullet Y| \leqslant \kappa/q\,.$$