Semidefinite programs and combinatorial optimization

Lecture notes by L. Lovász
Microsoft Research
Redmond, WA 98052
lovasz@microsoft.com
http://www.research.microsoft.com/ lovasz

Contents

1 Introduction

Linear programming has been one of the most fundamental and successful tools in optimization and discrete mathematics. Its applications include exact and approximation algorithms, as well as structural results and estimates. The key point is that linear programs are very efficiently solvable, and have a powerful duality theory.

A fundamental method in combinatorial optimization is to write a combinatorial optimization problem as a linear program with integer variables. There are usually many ways to do so; ideally, one tries to get the "tightest" description (in which the feasible set of the linear program is the convex hull of integer solutions); but this is often too complicated to determine, and we have to work with a "relaxed" description. We then forget the integrality constraints, thereby obtaining a *linear relaxation*, a linear program which can be solved efficiently; and then trying to restore the integrality of the variables by some kind of rounding (which is usually heuristic, and hence only gives approximately optimal integral solutions). In those particularly well-structured cases when we have the tightest description, the basic optimal solutions to the linear program are automatically integral, so it also solves the combinatorial optimization problem right away.

Linear programs are special cases of convex programs; *semidefinite programs* are more general but still convex programs, to which many of the useful properties of linear programs extend. Recently, semidefinite programming arose as a generalization of linear programming with substantial novel applications. Again, it can be used both in proofs and in the design of exact and approximation algorithms. It turns out that various combinatorial optimization problems have semidefinite (rather than linear) relaxations which are still efficiently computable, but approximate the optimum much better. This fact has lead to a real breakthrough in approximation algorithms.

In these notes we survey semidefinite optimization mainly as a relaxation of discrete optimization problems. We start with two examples, a proof and an approximation algorithm, where semidefinite optimization plays a important role. Still among the preliminaries, we survey some areas which play a role in semidefinite optimization: linear algebra (in particular, positive semidefinite matrices), linear programming (duality and algorithms), and polyhedral combinatorics (which we illustrate on the example of the stable set polytope).

After introducing semidefinite programs and discussing some of their basic properties, we show that semidefinite programs arise in a variety of ways: as certain geometric extremal problems, as relaxations (stronger than linear relaxations) of combinatorial optimization problems, in optimizing eigenvalue bounds in graph theory, as stability problems in engineering.

Next we show through examples from graph theory, number theory, and logic how semidefinite optimization can be used in proofs as well as in the design of approximation algorithms.

In Chapter 7 we try to put the combinatorial applications of semidefinite optimization in a broader perspective: they can be viewed as procedures to strengthen the descriptions of combinatorial optimization problems as integer linear programs. It turns out that such procedures can be formalized, and in some cases (like the stable set polytope, our favorite example) they lead to efficient ways of generating the tight linear descriptions for most cases when this description is known at all.

There are many unsolved problems in this area; indeed, progress has been quite slow (but steady) due to the difficulty of some of those. Several of these roadblocks are described in Chapter 8.

For more comprehensive studies of issues concerning semidefinite optimization, see [98].

1.1 Shannon capacity

Consider a noisy channel through which we are sending messages over a finite alphabet V. The noise may blur some letters so that certain pairs can be confounded. We want to select as many words of length k as possible so that no two can possibly be confounded. As we shall see, the number of words we can select grows as Θ^k for some $\Theta \geq 1$, which is called the *Shannon zero-error capacity* of the channel.

In terms of graphs, we can model the problem as follows. We consider V as the set of nodes of a graph, and connect two of them by an edge if they can be confounded. This way we obtain a graph $G = (V, E)$. We denote by $\alpha(G)$ the maximum number of independent points (the maximum size of a stable set) in the graph G. If $k = 1$, then the maximum number of non-confoundable messages is $\alpha(G)$.

To describe longer messages, we define the *strong product* $G \cdot H$ of two graphs $G = (V, E)$ and $H = (W, F)$ as the graph with $V(G \cdot H) = V \times W$, with $(i, u)(j, v) \in E(G \cdot H)$ iff $ij \in E$ and $uv \in F$, or $ij \in E$ and $u = v$, or $i = j$ and $uv \in F$. The product of k copies of G is denoted by G^k. Thus $\alpha(G^k)$ is the maximum number of words of length k, composed of elements of V, so that for every two words there is at least one i ($1 \leq i \leq k$) such that the i-th letters are different and non-adjacent in G, i.e., non-confoundable.

The *Shannon capacity* of a graph G is the value $\Theta(G) = \lim_{k\to\infty} \alpha(G^k)^{1/k}$ (it is not hard to see that the limit exists). It is not known whether $\Theta(G)$ can be computed for all graphs by any algorithm (polynomial or not), although there are several special classes of graphs for which this is not hard. For example, if G is a 4-cycle with nodes (a, b, c, d), then for every $k \geq 1$, all words of length k consisting of a and c only can be used, and so $\alpha(C_4^k) \geq 2^k$. On the other hand, if we use a word, then all the 2^k words obtained from it by replacing a and b by each other, as well as c and d by each other, are excluded. Hence $\alpha(C_4^k) \leq 4^k/2^k = 2^k$, and $\Theta(C_4) = 2$. More generally, we have $\alpha(G^k) \geq \alpha(G)^k$ for any graph G and so $\Theta((G) \geq \alpha(G)$. If we can also bound $\Theta(G)$ from above by $\alpha(G)$, then we have determined it exactly (this method works for all perfect graphs; cf section 2.3).

The smallest graph for which $\Theta(G)$ cannot be computed by such elementary means is the pentagon C_5. If we set $V(C_5) = \{0, 1, 2, 3, 4\}$ with $E(C_5) = \{01, 12, 23, 34, 40\}$, then C_5^2 contains the stable set $\{(0, 0), (1, 2), (2, 4), (3, 1), (4, 3)\}$. So $\alpha(C_5^{2k}) \geq \alpha(C_5^2)^k \geq 5^k$, and hence $\Theta(C_5) \geq \sqrt{5}$.

We show that equality holds here [64]. Consider an "umbrella" in $\mathbb{R}^3$ with the unit vector e_1 as its handle, and 5 ribs of unit length (Figure 1). Open it up to the point when non-consecutive ribs are orthogonal (i.e., form an angle of 90°). This way we get 5 unit vectors u_0, u_1, u_2, u_3, u_4, assigned to the nodes of C_5 so that each u_i forms the same angle with e_1 and any two non-adjacent nodes are labeled with orthogonal vectors. (Elementary trigonometry gives $e_1^\top u_i = 5^{-1/4}$).

It turns out that we can obtain a similar labeling of the nodes of C_5^k by unit vectors $v_i \in \mathbb{R}^{3k}$, so that any two non-adjacent nodes are labeled with orthogonal vectors. Moreover,

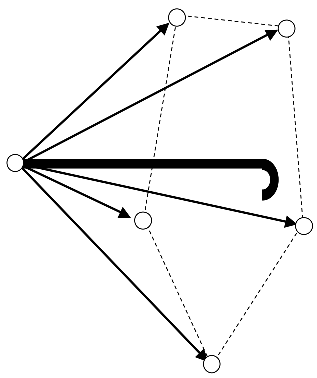

Figure 1: An orthogonal representation of C_5.

$e_1^\mathsf{T} v_i = 5^{-k/4}$ for every $i \in V(C_5^k)$. Such a labeling is obtained by taking tensor products. The *tensor product* of two vectors $(u_1, \dots, u_n) \in \mathbb{R}^n$ and $(v_1, \dots, v_m) \in \mathbb{R}^m$ is the vector

$$u \circ v = (u_1 v_1, \dots, u_1 v_m, u_2 v_1, \dots, u_2 v_m, \dots, u_n v_1, \dots, u_n v_m) \in \mathbb{R}^{nm}.$$

The tensor product of several vectors is defined similarly. The property one uses in verifying the properties claimed above is that if $u, x \in \mathbb{R}^n$ and $v, y \in \mathbb{R}^m$, then

$$(u \circ v)^\mathsf{T} (x \circ y) = (u^\mathsf{T} x)(v^\mathsf{T} y).$$

If S is any stable set in C_5^k, then $\{v_i : \ i \in S\}$ is a set of mutually orthogonal unit vectors, and hence

$$\sum_{i \in S} (e_1^\mathsf{T} v_i)^2 \le |e_1|^2 = 1$$

(if the v_i formed a basis then this inequality would be an equality).

On the other hand, each term on the left hand side is $5^{-1/4}$, hence the left hand side is $|S| 5^{-k/2}$, and so $|S| \le 5^{k/2}$. Thus $\alpha(C_5^k) \le 5^{k/2}$ and $\Theta(C_5) = \sqrt{5}$.

This method extends to any graph $G = (V, E)$ in place of C_5: all we have to do is to assign unit vectors to the nodes so that non-adjacent nodes correspond to orthogonal vectors (such an assignment will be called an *orthogonal representation*). If the first coordinate of each of these vectors is s, then the Shannon capacity of the graph is at most $1/s^2$. The best bound that can be achieved by this method will be denoted by $\vartheta(G)$.

But how to construct an optimum (or even good) orthogonal representation? Somewhat surprisingly, the optimum representation can be computed in polynomial time using semidefinite optimization. Furthermore, it has many nice properties, most of which are derived using semidefinite duality and other fundamental properties of semidefinite programs (section 3.1), as we shall see in section 5.1.

1.2 Maximum cuts

A *cut* in a graph $G = (V, E)$ is the set of edges connecting a set $S \subseteq V$ to $V \setminus S$, where $\emptyset \subset S \subset V$. The *Max Cut Problem* is to find a cut with maximum cardinality. We denote by MC this maximum.

(More generally, we can be given a weighting $w: V \to \mathbb{R}_+$, and we could be looking for a cut with maximum total weight. Most other problems discussed below, like the stable set problem, have such weighted versions. To keep things simple, however, we usually restrict our discussions to the unweighted case.)

The Max Cut Problem is NP-hard; one natural approach is to find an approximately maximum cut. Formulated differently, Erdős in 1967 described the following simple heuristic: for an arbitrary ordering $(v_1, \ldots, v_n)$ of the nodes, we color $v_1, v_2, \ldots, v_n$ successively red or blue. For each i, v_i is colored blue iff the number of edges connecting v_i to blue nodes among $v_1, \ldots, v_{i-1}$ is less than the number of edges connecting v_i to red nodes in this set. Then the cut formed by the edges between red and blue nodes contains at least half of all edges. In particular, we get a cut that is at least half as large as the maximum cut.

There is an even easier randomized algorithm to achieve this approximation, at least in expected value. Let us 2-color the nodes of G randomly, so that each node is colored red or blue independently, with probability 1/2. Then the probability that an edge belongs to the cut between red and blue is 1/2, and expected number of edges in this cut is $|E|/2$.

Both of these algorithms show that the maximum cut can be approximated from below in polynomial time with a multiplicative error of at most 1/2. Can we do better? The following strong negative result [10, 19, 45] shows that we cannot get arbitrarily close to the optimum:

Proposition 1.1 *It is NP-hard to find a cut with more than* $(16/17)\mathcal{MC} \approx .94\mathcal{MC}$ *edges.*

But we can do better than 1/2, as the following seminal result of Goemans and Williamson [37, 38] shows:

Theorem 1.2 *One can find in polynomial time a cut with at least* $.878\mathcal{MC}$ *edges.*

The algorithm of Goemans and Williamson makes use of the following geometric construction. We want to find an embedding $i \mapsto u_i$ $(i \in V)$ of the nodes of the graph in the unit sphere in $\mathbb{R}^d$ so that the following "energy" is minimized:

$$\mathcal{E} = -\sum_{ij \in E} \frac{1}{4}(u_i - u_j)^2 = -\sum_{ij \in E} \frac{1 - u_i^\mathsf{T} u_j}{2}.$$

(Note the negative sign: this means that the "force" between adjacent nodes is repulsive, and grows linearly with the distance.)

If we work in $\mathbb{R}^1$, then the problem is equivalent to MAX CUT: each node is represented by either 1 or -1, and the edges between differently labeled nodes contribute -1 to the energy, the other edges contribute 0. Hence the negative of the minimum energy $\mathcal{E}$ is an upper bound on the maximum size $\mathcal{MC}$ of a cut.

Unfortunately, the argument above also implies that for $d = 1$, the optimal embedding is NP-hard to find. While I am not aware of a proof of this, it is probably NP-hard for $d = 2$ and more generally, for any fixed d. The surprising fact is that for $d = n$, such an embedding can be found using semidefinite optimization (cf. section 4.1).

So $-\mathcal{E}$ is a polynomial time computable upper bound on the size of the maximum cut. How good is this bound? And how to construct an approximately optimum cut from this representation? Here is the simple but powerful trick: *take a random hyperplane H through* *the origin in* $\mathbb{R}^n$ (Figure 2). The partition of R^d given by H yields a cut in our graph. Since the construction pushes adjacent points apart, one expects that the random cut will intersect many edges.

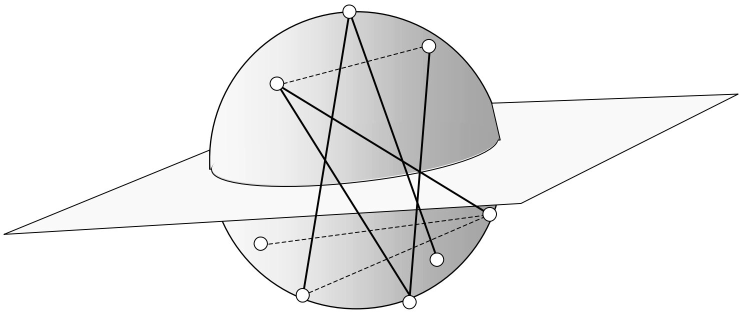

Figure 2: A cut in the graph given by a random hyperplane

To be more precise, let $ij \in E$ and let $u_i, u_j \in S^{n-1}$ be the corresponding vectors in the embedding constructed above. It is easy to see that the probability that a random hyperplane H through 0 separates u_i and u_j is α_{ij}/π, where $\alpha_{ij} = \arccos u_i^\mathsf{T} u_j$ is the angle between u_i and u_j. It is not difficult to verify that if $-1 \leq t \leq 1$, then $\arccos t \geq 1.38005(1-t)$. Thus the expected number of edges intersected by H is

$$\sum_{ij \in E} \frac{\arccos u_i^\mathsf{T} u_j}{\pi} \geq \sum_{ij \in E} 1.38005 \frac{1 - u_i^\mathsf{T} u_j}{\pi} = \frac{1.38005}{\pi} 2(-\mathcal{E}) \geq .878 \mathcal{MC}.$$

(One objection to the above algorithm could be that it uses random numbers. In fact, the algorithm can be *derandomized* by well established but non-trivial techniques. We do not consider this issue in these notes; see e.g. [5], Chapter 15 for a survey of derandomization methods.)

2 Preliminaries

We collect some of the basic results from linear programming, linear algebra, and polyhedral combinatorics that we will use. While this is all textbook material, it will be convenient to have this collection of results for the purposes of notation, reference and comparison. [88] is a reference for linear algebra, and a [79], for linear programming.

2.1 Linear algebra

As the title of these lecture notes suggests, we'll be concerned with semidefinite matrices; to get to these, we start with a review of eigenvalues, and in particular eigenvalues of symmetric matrices.

Let A be an $n \times n$ real matrix. An *eigenvector* of A is a vector such that Ax is parallel to x; in other words, $Ax = \lambda x$ for some real or complex number λ. This number λ is called the *eigenvalue* of A belonging to eigenvector v. Clearly λ is an eigenvalue iff the matrix

$A - \lambda I$ is singular, equivalently, iff $det(A - \lambda I) = 0$. This is an algebraic equation of degree n for λ, and hence has n roots (with multiplicity).

The *trace* of the (square) matrix $A = (A_{ij})$ is defined as

$$\mathrm{tr}(A) = \sum_{i=1}^{n} A_{ii}.$$

The trace of A is the sum of the eigenvalues of A, each taken with the same multiplicity as it occurs among the roots of the equation $det(A - \lambda I) = 0$.

If the matrix A is symmetric, then its eigenvalues and eigenvectors are particularly well behaved. All the eigenvalues are real. Furthermore, there is an orthogonal basis $v_1, \ldots, v_n$ of the space consisting of eigenvectors of A, so that the corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$ are precisely the roots of $det(A - \lambda I) = 0$. We may assume that $|v_1| = \ldots = |v_n| = 1$; then A can be written as

$$A = \sum_{i=1}^{n} \lambda_i v_i v_i^{\mathsf{T}}.$$

Another way of saying this is that every symmetric matrix can be written as $U^{\mathsf{T}} D U$, where U is an orthogonal matrix and D is a diagonal matrix. The eigenvalues of A are just the diagonal entries of D.

To state a further important property of eigenvalues of symmetric matrices, we need the following definition. A *symmetric minor* of A is a submatrix B obtained by deleting some rows and the *corresponding* columns.

Theorem 2.1 (Interlacing eigenvalues) *Let A be an $n \times n$ symmetric matrix with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n$. Let B be an $(n-k) \times (n-k)$ symmetric minor of A with eigenvalues $\mu_1 \geq \ldots \geq \mu_{n-k}$. Then*

$$\lambda_i \leq \mu_i \leq \lambda_{i+k}.$$

Now we come to the definition that is crucial for our lectures. A symmetric $n \times n$ matrix A is called *positive semidefinite*, if all of its eigenvalues are nonnegative. This property is denoted by $A \succeq 0$. The matrix is *positive definite*, if all of its eigenvalues are positive.

There are many equivalent ways of defining positive semidefinite matrices, some of which are summarized in the Proposition below.

Proposition 2.2 *For a real symmetric $n \times n$ matrix A, the following are equivalent:*

(i) *A is positive semidefinite;*

(ii) *the quadratic form $x^T A x$ is nonnegative for every $x \in \mathbb{R}^n$;*

(iii) *A can be written as the Gram matrix of n vectors $u_1, \ldots, u_n \in \mathbb{R}^m$ for some m; this means that $a_{ij} = u_i^{\mathsf{T}} u_j$. Equivalently, $A = U^{\mathsf{T}} U$ for some matrix U;*

(iv) *A is a nonnegative linear combination of matrices of the type xx^{T};*

(v) *The determinant of every symmetric minor of A is nonnegative.*

Let me add some comments. The least m for which a representation as in (iii) is possible is equal to the rank of A. It follows e.g. from (ii) that the diagonal entries of any positive semidefinite matrix are nonnegative, and it is not hard to work out the case of equality: all entries in a row or column with a 0 diagonal entry are 0 as well. In particular, the trace of a positive semidefinite matrix A is nonnegative, and $\mathrm{tr}(A) = 0$ if and only if $A = 0$.

The sum of two positive semidefinite matrices is again positive semidefinite (this follows e.g. from (ii) again). The simplest positive semidefinite matrices are of the form aa^{T} for some vector a (by (ii): we have $x^{\mathsf{T}}(aa^{\mathsf{T}})x = (a^{\mathsf{T}}x)^2 \geq 0$ for every vector x). These matrices are precisely the positive semidefinite matrices of rank 1. Property (iv) above shows that every positive semidefinite matrix can be written as the sum of rank-1 positive semidefinite matrices.

The product of two positive semidefinite matrices A and B is not even symmetric in general (and so it is not positive semidefinite); but the following can still be claimed about the product:

Proposition 2.3 *If A and B are positive semidefinite matrices, then $\mathrm{tr}(AB) \geq 0$, and equality holds iff $AB = 0$.*

Property (v) provides a way to check whether a given matrix is positive semidefinite. This works well for small matrices, but it becomes inefficient very soon, since there are many symmetric minors to check. An efficient method to test if a symmetric matrix A is positive semidefinite is the following algorithm. Carry out 2-sided Gaussian elimination on A, pivoting always on diagonal entries ("2-sided" means that we eliminate all entries in both the row and the column of the pivot element).

If you ever find a negative diagonal entry, or a 0 diagonal entry whose row contains a non-zero, stop: the matrix is not positive semidefinite. If you obtain an all-zero matrix (or eliminate the whole matrix), stop: the matrix is positive semidefinite.

If this simple algorithm finds that A is not positive semidefinite, it also provides a certificate in the form of a vector v with $v^{\mathsf{T}}Av < 0$. Assume that the i-th diagonal entry of the matrix $A^{(k)}$ after k steps is negative. Write $A^{(k)} = E_k^{\mathsf{T}} \dots E_1^{\mathsf{T}} A E_1 \dots E_k$, where E_i are elementary matrices. Then we can take the vector $v = E_1 \dots E_k e_i$. The case when there is a 0 diagonal entry whose row contains a non-zero is similar.

It will be important to think of $n \times n$ matrices as vectors with n^2 coordinates. In this space, the usual inner product is written as $A \cdot B$. This should not be confused with the matrix product AB. However, we can express the inner product of two $n \times n$ matrices A and B as follows:

$$A \cdot B = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} B_{ij} = \mathrm{tr}(A^{\mathsf{T}} B).$$

Positive semidefinite matrices have some important properties in terms of the geometry of this space. To state these, we need two definitions. A *convex cone* in $\mathbb{R}^n$ is a set of vectors which along with any vector, also contains any positive scalar multiple of it, and along with any two vectors, also contains their sum. Any system of homogeneous linear inequalities

$$a_1^{\mathsf{T}} x \geq 0, \quad \dots \quad a_m^{\mathsf{T}} x \geq 0$$

defines a convex cone; convex cones defined by such (finite) systems are called *polyhedral*.

For every convex cone C, we can form its *polar cone* C^*, defined by

$$C^* = \{x \in \mathbb{R}^n : x^\mathsf{T} y \geq 0 \ \forall y \in C\}.$$

This is again a convex cone. If C is closed (in the topological sense), then we have $(C^*)^* = C$.

The fact that the sum of two such matrices is again positive semidefinite (together with the trivial fact that every positive scalar multiple of a positive semidefinite matrix is positive semidefinite), translates into the geometric statement that *the set of all positive semidefinite matrices forms a convex closed cone* P_n *in* $\mathbb{R}^{n\times n}$ *with vertex* 0. This cone P_n is important, but its structure is quite non-trivial. In particular, it is non-polyhedral for $n \geq 2$; for $n = 2$ it is a nice rotational cone (Figure 3; the fourth coordinate x_{21}, which is always equal to x_{12} by symmetry, is suppressed). For $n \geq 3$ the situation becomes more complicated, because P_n is neither polyhedral nor smooth: any matrix of rank less than $n-1$ is on the boundary, but the boundary is not differentiable at that point.

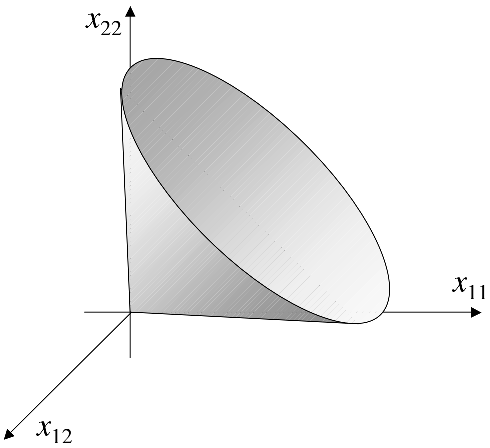

Figure 3: The semidefinite cone for $n = 2$.

The polar cone of $\mathcal{P}$ is itself; in other words,

Proposition 2.4 *A matrix A is positive semidefinite iff $A \cdot B \geq 0$ for every positive semidefinite matrix B.*

We conclude this little overview with a further basic fact about nonnegative matrices.

Theorem 2.5 (Perron-Frobenius) *If an $n\times n$ matrix has nonnegative entries then it has a nonnegative real eigenvalue λ which has maximum absolute value among all eigenvalues. This eigenvalue λ has a nonnegative real eigenvector. If, in addition, the matrix has no block-triangular decomposition (i.e., it does not contain a $k \times (n-k)$ block of 0-s disjoint from the diagonal), then λ has multiplicity 1 and the corresponding eigenvector is positive.*

2.2 Linear programming

Linear programming is closely related to (in a sense the same as) the study of systems of linear inequalities. At the roots of this theory is the following basic lemma.

Lemma 2.6 (Farkas Lemma) *A system of linear inequalities* $a_1^T x \le b_1, \ldots, a_m^T x \le b_m$ *has no solution iff there exist* $\lambda_1, \ldots, \lambda_m \ge 0$ *such that* $\sum_i \lambda_i a_i = 0$ *and* $\sum_i \lambda_i b_i = -1$.

Let us make a remark about the computational complexity aspect of this. The solvability of a system of linear inequalities is in NP ("just show the solution"; to be precise, one has to argue that there is a rational solution with small enough numerators and denominators so that it can be exhibited in space polynomial in the input size; but this can be done). One consequence of the Farkas Lemma (among many) is that this problem is also in co-NP ("just show the λ's").

A closely related statement is the following:

Lemma 2.7 (Farkas Lemma, inference version) *Let* $a_1, \ldots, a_m, c \in \mathbb{R}^n$ *and* $b_1, \ldots, b_m$, $d \in \mathbb{R}$. *Assume that the system* $a_1^T x \le b_1, \ldots, a_m^T x \le b_m$ *has a solution. Then* $c^T x \le d$ *for all solutions of* $a_1^T x \le b_1, \ldots, a_m^T x \le b_m$ *iff there exist* $\lambda_1, \ldots, \lambda_m \ge 0$ *such that* $c = \sum_i \lambda_i a_i$ *and* $d \ge \sum_i \lambda_i b_i$.

This again can be put into a general context: there is a semantical notion of a linear inequality being a consequence of others (it holds whenever the others do), and a syntactical (it is a linear combination of the others with non-negative coefficients). The lemma asserts that these two are equivalent. We'll see that e.g. for quadratic inequalities, the situation is more complicated.

Now we turn to linear programming. A typical linear program has the following form.

$$\begin{array}{lrcl} \text{maximize} & c^{\mathsf{T}} x & & \\ \text{subject to} & a_1^{\mathsf{T}} x & \le & b_1, \\ & & \vdots & \\ & a_m^{\mathsf{T}} x & \le & b_m, \end{array} \tag{1}$$

where $a_1, \ldots, a_m$ are given vectors in $\mathbb{R}^n$, $b_1, \ldots, b_m$ are real numbers, and $x = (x_1, \ldots, x_n)$ is a vector of n unknowns. These inequalities can be summed up in matrix form as $Ax \le b$, where A is a matrix with m rows and m columns and $b \in \mathbb{R}^m$.

It is very fruitful to think of linear programs geometrically. The solution of the constraint system $Ax \le b$ (also called *feasible solutions*) form a convex polyhedron P in $\mathbb{R}^n$. For the following discussion, let us assume that P is bounded and has an internal point. Then each facet ($(n-1)$-dimensional faces) of P corresponds to one of the inequalities $a_i^{\mathsf{T}} x \le b_i$ (there may be other inequalities in the system, but those are redundant). The *objective function* $c^{\mathsf{T}} x$ can be visualized as a family of parallel hyperplanes; to find its maximum over P means to translate this hyperplane in the direction of the vector c as far as possible so that it still intersects P. If P is bounded, then these "ultimate" common points will form a face (a vertex, an edge, or higher dimensional face) P, and there will be at least one vertex among them (see Figure 4).

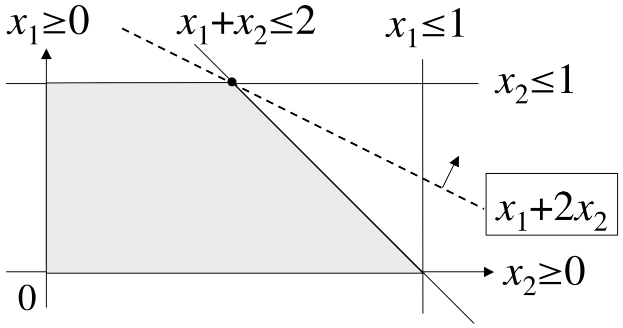

Figure 4: The feasible domain and optimum solution of the linear program: maximize $x_1 + 2x_2$, subject to $0 \leq x_1 \leq 2$, $0 \leq x_2 \leq 1$, and $x_1 + x_2 \leq 2$.

There are many alternative ways to describe a linear program. We may want to maximize instead of minimize; we may have equations, and/or inequalities of the form $\geq$. Sometimes we consider only nonnegative variables; the inequalities $x_i \geq 0$ may be included in (1), but it may be advantageous to separate them. All these versions are easily reduced to each other.

The *dual* of (1) is the linear program

$$\begin{array}{lrcl} \text{minimize} & b^{\mathsf{T}} y & & \\ \text{subject to} & A^{\mathsf{T}} y & = & c, \\ & y & \geq & 0. \end{array} \tag{2}$$

The crucial property of this very important construction is the following.

Theorem 2.8 (Duality Theorem) *If either one of the primal and dual programs has an optimum solution, then so does the other and the two optimum values are equal.*

The primal program is infeasible if and only if the dual is unbounded. The dual program is infeasible iff the primal is unbounded.

The primal and dual linear programs are related to each other in many ways. The following theorem describes the relationship between their optimal solutions.

Theorem 2.9 (Complementary Slackness Theorem) *Let x be a solution of the primal program and y, a solution of the dual program. Then both x and y are optimal if and only if for every j with $y_j > 0$, the j-th constraint of the primal problem (1) is satisfied with equality.*

Linear programs are solvable in polynomial time. The classical, and still very well usable algorithm to solve them is the *Simplex Method*. This is practically quite efficient, but can be exponential on some instances. The first polynomial time algorithm to solve linear programs was the *Ellipsoid Method*; this is, however, impractical. The most efficient methods known today, both theoretically and practically, are *Interior Point Methods*.

2.3 Polyhedral combinatorics: the stable set polytope

The basic technique of applying linear programming in discrete optimization is polyhedral combinatorics. Instead of surveying this broad topic, we illustrate it by recalling some results on the stable set polytope. A detailed account can be found e.g. in [43].

Let $G = (V, E)$ be a graph; it is convenient to assume that it has no isolated nodes. The *Stable Set Problem* is the problem of finding $\alpha(G)$. This problem is NP-hard.

The basic idea in applying linear programming to study the stable set problem is the following. For every subset $S \subseteq V$, let $\chi^S \in \mathbb{R}^V$ denote its incidence vector, i.e., the vector defined by

$$\chi_i^S = \begin{cases} 1, & \text{if } i \in S, \\ 0, & \text{otherwise,} \end{cases}$$

The *stable set polytope* STAB(G) of G is the convex hull of incidence vectors of all stable sets.

There is a system of linear inequalities whose solution set is exactly the polytope STAB(G), and if we can find this system, then we can find $\alpha(G)$ by optimizing the linear objective function $\sum_i x_i$. Unfortunately, this system is in general exponentially large and very complicated. But if we can find at least some linear inequalities valid for the stable set polytope, then using these we get an upper bound on $\alpha(G)$, and for special graphs, we get the exact value.

Let us survey some classes of known constraints.

NON-NEGATIVITY CONSTRAINTS:

$$x_i \geq 0 \quad (i \in V). \tag{3}$$

EDGE CONSTRAINTS:

$$x_i + x_j \leq 1 \quad (ij \in E). \tag{4}$$

These inequalities define a polytope FSTAB(G). The integral points in FSTAB(G) are exactly the incidence vectors of stable sets, but FSTAB(G) may have other non-integral vertices, and is in general larger than STAB(G) (see Figure 5).

Proposition 2.10 (a) STAB(G) = FSTAB(G) *iff G is bipartite.*

(b) *The vertices of* FSTAB(G) *are half-integral.*

A *clique* is a maximal complete subgraph.

CLIQUE CONSTRAINTS:

$$\sum_{i \in B} x_i \leq 1, \quad \text{where } B \text{ is a clique.} \tag{5}$$

Inequalities (3) and (5) define a polytope QSTAB(G), which is contained in FSTAB(G), but is in general larger than STAB(G).

A graph G is called *perfect* if $\chi(G') = \omega(G')$ for every induced subgraph G' of G. If G is perfect then so is $\overline{G}$ [62]. See [39, 43, 65] for the theory of perfect graphs.

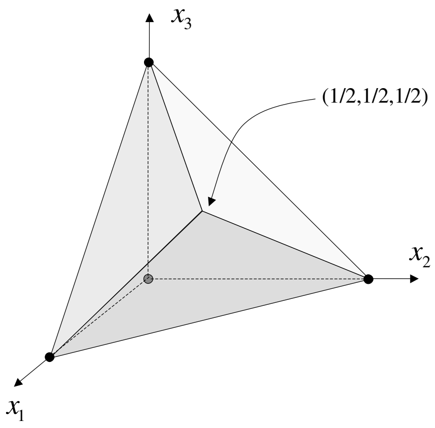

Figure 5: The fractional stable set polytope of the triangle. The black dots are incidence vectors of stable sets; the vertex $(1/2, 1/2, 1/2)$ (closest to us) is not a vertex of $\mathrm{STAB}(K_3)$.

Theorem 2.11 [Fulkerson–Chvatal] *$STAB(G) = \mathrm{QSTAB}(G)$ iff G is perfect.*

A *convex corner* in $\mathbb{R}^V$ is a full-dimensional, compact, convex set P such that $x \in P$, $0 \le y \le x$ implies $y \in P$. The *antiblocker* of a convex corner P is defined as $P^* = \{x \in \mathbb{R}_+^V : x^\mathsf{T} y \le 1 \text{ for all } y \in P\}$. P^* is a convex corner and $P^{**} = P$. Figure 6 illustrates this important notion in 2 dimensions.

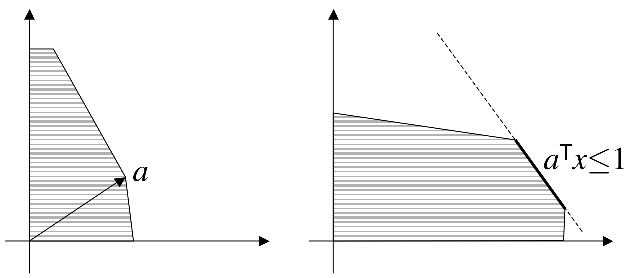

Figure 6: A pair of antiblocking convex corners. The vertex a on the left corresponds to the facet $ax \le 1$ on the right.

Proposition 2.12 *For every graph G,*

$$\mathrm{QSTAB}(G) = \mathrm{STAB}(\overline{G})^*.$$

ODD HOLE CONSTRAINTS:

$$\sum_{i \in C} x_i \le \frac{|C| - 1}{2}, \quad \text{where } C \text{ induces a cordless odd cycle.} \tag{6}$$

A graph is called *t-perfect* if (3), (4) and (6) suffice to describe $\mathrm{STAB}(G)$, and *h-perfect* if (3), (5) and (6) suffice to describe $\mathrm{STAB}(G)$.

Odd antihole constraints:

$$\sum_{i \in B} x_i \leq 2, \quad \text{where } B \text{ induces the complement of a cordless odd cycle.} \tag{7}$$

How strong are these inequalities? An inequality valid for a (for simplicity, full-dimensional) polytope $P \subseteq \mathbb{R}^n$ is called a *facet* if there are n affine independent vertices of P that satisfy it with equality. Such an inequality must occur in every description of P by linear inequalities (up to scaling by a positive number). The clique constraints are all facets, the odd hole and antihole inequalities are facets if $B = V$, and in many other cases. (If there are nodes not occurring in the inequality then they may sometimes be added to the constraint with non-zero coefficient; this is called *lifting*.)

All the previous constraints are special cases of the following construction. Let G_S denote the subgraph of G induced by $S \subseteq V$.

Rank constraints:

$$\sum_{i \in S} x_i \leq \alpha(G_S).$$

For this general class of constraints, however, we cannot even compute the right hand side efficiently. Another of their shortcomings is that we don't know when they are facets (or can be lifted to facets). An important special case when at least we know that they are facets was described by Chvátal [23]. A graph G is called α-critical if it has no isolated nodes, and deleting any edge e, $\alpha(G)$ increases. These graphs have an interesting and non-trivial structure theory; here we can only include figure 7 showing some of them.

Theorem 2.13 *Let $G = (V, E)$ be an α-critical graph. Then the inequality $\sum_{i \in V} x_i \leq \alpha(G)$ defines a facet of* $\mathrm{STAB}(G)$.

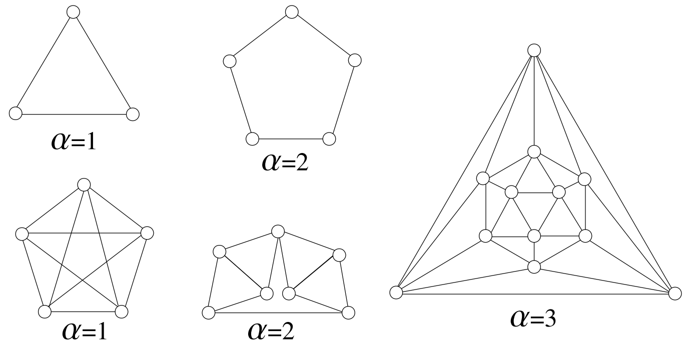

Figure 7: Some α-critical graphs.

3 Semidefinite programs

A semidefinite program is an optimization problem of the following form:

$$\begin{array}{ll} \text{minimize} & c^{\mathsf{T}} x \\ \text{subject to} & x_1 A_1 + \ldots x_n A_n - B \succeq 0 \end{array} \tag{8}$$

Here $A_1, \ldots, A_n, B$ are given symmetric $m \times m$ matrices, and $c \in \mathbb{R}^n$ is a given vector. We can think of $X = x_1 A_1 + \ldots x_n A_n - B$ as a matrix whose entries are linear functions of the variables.

As usual, any choice of the values x_i that satisfies the given constraint is called a *feasible solution*. A solution is *strictly feasible*, if the matrix X is positive definite. We denote by v_{primal} the supremum of the objective function.

The special case when $A_1, \ldots, A_n, B$ are diagonal matrices is just a "generic" linear program, and it is very fruitful to think of semidefinite programs as generalizations of linear programs. But there are important technical differences. The following example shows that, unlike in the case of linear programs, the supremum may be finite but not a maximum, i.e., not attained by any feasible solution.

Example 3.1 Consider the semidefinite program

$$\begin{array}{ll} \text{minimize} & x_1 \\ \text{subject to} & \begin{pmatrix} x_1 & 1 \\ 1 & x_2 \end{pmatrix} \succeq 0 \end{array}$$

The semidefiniteness condition boils down to the inequalities $x_1, x_2 \geq 0$ and $x_1 x_2 \geq 1$, so the possible values of the objective function are all positive real numbers. Thus $v_{\text{primal}} = 0$, but the supremum is not attained.

As in the theory of linear programs, there are a large number of equivalent formulations of a semidefinite program. Of course, we could consider minimization instead of maximization. We could stipulate that the x_i are nonnegative, or more generally, we could allow additional linear constraints on the variables x_i (inequalities and/or equations). These could be incorporated into the form above by extending the A_i and B with new diagonal entries.

We could introduce the entries of A as variables, in which case the fact that they are linear functions of the original variables translates into linear relations between them. Straightforward linear algebra transforms (8) into an optimization problem of the form

$$\begin{array}{llcl} \text{maximize} & C \cdot X & & \\ \text{subject to} & X & \succeq & 0 \\ & D_1 \cdot X & = & d_1 \\ & & \vdots & \\ & D_k \cdot X & = & d_k, \end{array} \tag{9}$$

where $C, D_1, \ldots, D_k$ are symmetric $m \times m$ matrices and $d_1, \ldots, d_k \in \mathbb{R}$. Note that $C \cdot X$ is the general form of a linear combination of entries of X, and so $D_i \cdot X = d_i$ is the general form of a linear equation in the entries of X.

It is easy to see that we would not get any substantially more general problem if we allowed linear inequalities in the entries of X in addition to the equations.

3.1 Fundamental properties of semidefinite programs

We begin with the semidefinite version of the Farkas Lemma:

Lemma 3.2 [Homogeneous version] *Let $A_1, \ldots, A_n$ be symmetric $m \times m$ matrices. The system*

$$x_1 A_1 + \ldots + x_n A_n \succ 0$$

has no solution in $x_1, \ldots, x_n$ if and only if there exists a symmetric matrix $Y \neq 0$ such that

$$\begin{aligned} A_1 \cdot Y &= 0 \\ A_2 \cdot Y &= 0 \\ &\vdots \\ A_n \cdot Y &= 0 \\ Y &\succeq 0\,. \end{aligned}$$

Proof. As discussed in section 2.1, the set $\mathcal{P}_m$ of $m \times m$ positive semidefinite matrices forms a closed convex cone. If

$$x_1 A_1 + \ldots + x_n A_n \succ 0$$

has no solution, then the linear subspace L of matrices of the form $x_1 A_1 + \ldots x_n A_n$ is disjoint from the interior of this cone PSD^m. It follows that this linear space is contained in a hyperplane that is disjoint from the interior of PSD^m. This hyperplane can be described as $\{X : Y \cdot X = 0\}$, where we may assume that $X \cdot Y \geq 0$ for every $X \in PSD^m$. Then $Y \neq 0$, $Y \succeq 0$ by Lemma 2.4, and $A_i \cdot Y = 0$ since the A_i belong to L. □

By similar methods one can prove:

Lemma 3.3 [Inhomogeneous version] *Let $A_1, \ldots, A_n, B$ be symmetric $m \times m$ matrices. The system*

$$x_1 A_1 + \ldots x_n A_n - B \succ 0$$

has no solution in $x_1, \ldots, x_n$ if and only if there exists a symmetric matrix $Y \neq 0$ such that

$$\begin{aligned} A_1 \cdot Y &= 0 \\ A_2 \cdot Y &= 0 \\ &\vdots \\ A_n \cdot Y &= 0 \\ B \cdot Y &\geq 0 \\ Y &\succeq 0\,. \end{aligned}$$

Given a semidefinite program (8), one can formulate the *dual program*:

$$\begin{array}{lrcl} \text{maximize} & B \cdot Y & & \\ \text{subject to} & A_1 \cdot Y & = & c_1 \\ & A_2 \cdot Y & = & c_2 \\ & & \vdots & \\ & A_n \cdot Y & = & c_m \\ & Y & \succeq & 0. \end{array} \tag{10}$$

Note that this too is a semidefinite program in the general sense. We denote by v_{dual} the supremum of the objective function.

With this notion of duality, the Duality Theorem holds in the following sense (see e.g. [96, 93, 94]):

Theorem 3.4 *Assume that both the primal and the dual semidefinite programs have feasible solutions. Then* $v_{\text{primal}} \ge v_{\text{dual}}$. *If, in addition, the primal program (say) has a strictly feasible solution, then the dual optimum is attained and* $v_{\text{primal}} = v_{\text{dual}}$. *In particular, if both programs have strictly feasible solutions, then the supremum resp. infimum of the objective functions are attained.*

Proof. Let $x_1, \ldots, x_n$ be any solution of (8) and Y, any solution of (10). By Proposition 2.3, we have

$$\sum_i c_i x_i - B \cdot Y = \operatorname{tr}(Y(\sum_i x_i A_i - B)) \ge 0,$$

which shows that $v_{\text{primal}} \ge v_{\text{dual}}$.
Moreover, the system

$$\begin{aligned} \sum_i c_i x_i &< v_{\text{primal}} \\ \sum_i x_i A_i &\succeq B \end{aligned}$$

has no solution in the x_i, by the definition of v_{primal}. Thus if we define the matrices

$$A_i' = \begin{pmatrix} -c_i & 0 \\ 0 & A_i \end{pmatrix}, \qquad B' = \begin{pmatrix} -v_{\text{primal}} & 0 \\ 0 & B \end{pmatrix},$$

then the system

$$x_1 A_1' + \ldots x_n A_n' - B' \succ 0$$

has no solution. By Lemma 3.3, there is a positive semidefinite matrix $Y' \ne 0$ such that

$$A_i' \cdot Y' = 0 \quad (i = 1, \ldots, n), \qquad B' \cdot Y' \ge 0.$$

Writing

$$Y' = \begin{pmatrix} y_{00} & y^{\mathsf{T}} \\ y & Y \end{pmatrix},$$

we get that

$$A_i \cdot Y = y_{00} c_i \quad (i = 1, \ldots, n), \qquad B \cdot Y \ge y_{00} v_{\text{primal}}.$$

We claim that $y_{00} \ne 0$. Indeed, if $y_{00} = 0$, then $y = 0$ by the semidefiniteness of Y', and since $Y' \ne 0$, it follows that $Y \ne 0$. The existence of Y would imply (by Lemma 3.3 again) that $x_1 A_1 + \ldots x_n A_n - B \succ 0$ is not solvable, which is contrary to the hypothesis about the existence of a strictly feasible solution.

Thus $y_{00} \neq 0$, and clearly $y_{00} > 0$. By scaling, we may assume that $y_{00} = 1$. But then Y is a feasible solution of the dual problem (10), with objective value $B \cdot Y \geq v_{\text{primal}}$, proving that $v_{\text{dual}} \geq v_{\text{primal}}$, and completing the proof.

The following *complementary slackness conditions* also follow from this argument.

Proposition 3.5 *Let x be a feasible solution of the primal program and Y, a feasible solution of the dual program. Then $v_{\text{primal}} = v_{\text{dual}}$ and both x and Y are optimal solutions if and only if $Y(\sum_i x_i A_i - B) = 0$.*

The following example shows that the somewhat awkward conditions about the strictly feasible solvability of the primal and dual programs cannot be omitted (see [83] for a detailed discussion of conditions for exact duality).

Example 3.6 Consider the semidefinite program

$$\begin{array}{ll} \text{minimize} & x_1 \\ \text{subject to} & \begin{pmatrix} 0 & x_1 & 0 \\ x_1 & x_2 & 0 \\ 0 & 0 & x_1 + 1 \end{pmatrix} \succeq 0 \end{array}$$

The feasible solutions are $x_1 = 0$, $x_2 \geq 0$. Hence v_{primal} is assumed and is equal to 0. The dual program is

$$\begin{array}{lrcl} \text{maximize} & -Y_{33} & & \\ \text{subject to} & Y_{12} + Y_{21} + Y_{33} & = & 1 \\ & Y_{22} & = & 0 \\ & Y & \succeq & 0\,. \end{array}$$

The feasible solutions are all matrices of the form

$$\begin{pmatrix} a & 0 & b \\ 0 & 0 & 0 \\ b & 0 & 1 \end{pmatrix}$$

where $a \geq b^2$. Hence $v_{\text{dual}} = -1$.

3.2 Algorithms for semidefinite programs

There are two essentially different algorithms known that solve semidefinite programs in polynomial time: the *ellipsoid method* and *interior point/barrier methods*. Both of these have many variants, and the exact technical descriptions are quite complicated; so we restrict ourselves to describing the general principles underlying these algorithms, and to some comments on their usefulness. We ignore numerical problems, arising from the fact that the optimum solutions may be irrational and the feasible regions may be very small; we refer to [82, 83] for discussions of these problems.

The first polynomial time algorithm to solve semidefinite optimization problems in polynomial time was the ellipsoid method. Let K be a convex body (closed, compact, convex, full-dimensional set) in $\mathbb{R}^N$. We set $S(K, t) = \{x \in \mathbb{R}^N : \; d(x, K) \leq t\}$, where d denotes euclidean distance. Thus $S(0, t)$ is the ball with radius t about 0.

A *(weak) separation oracle* for a convex body $K \subseteq \mathbb{R}^N$ is an oracle (a subroutine which is handled as a black box; one call on the oracle is counted as one step only) whose input is a rational vector $x \in \mathbb{R}^N$ and a rational $\varepsilon > 0$; the oracle either asserts that $x \in S(K, \varepsilon)$ or returns an "almost separating hyperplane" in the form of a vector $0 \neq y \in \mathbb{R}^N$ such that $y^\mathsf{T} x > y^\mathsf{T} z - \varepsilon |y|$ for all $z \in K$.

If we have a weak separation oracle for a convex body (in practice, any subroutine that realizes this oracle) then we can use the ellipsoid method to optimize any linear objective function over K [43]:

Theorem 3.7 *Let K be a convex body in $\mathbb{R}^n$ and assume that we know two real numbers $R > r > 0$ such that $S(0, r) \subseteq K \subseteq S(0, R)$. Assume further that we have a weak separation oracle for K. Let a (rational) vector $c \in \mathbb{R}^n$ and an error bound $0 < \varepsilon < 1$ be also given. Then we can compute a (rational) vector $x \in \mathbb{R}^n$ such that $x \in K$ and $c^\mathsf{T} x \geq c^T z - \varepsilon$ for every $y \in K$. The number of calls on the oracle and the number of arithmetic operations in the algorithm are polynomial in* $\log(R/r) + \log(1/\varepsilon) + n$.

This method can be applied to solve semidefinite programs in polynomial time, modulo some technical conditions. (Note that some complications arise already from the fact that the optimum value is not necessarily a rational number, even if all parameters are rational. A further warning is example 3.6.)

Assume that we are given a semidefinite program (8) with rational coefficients and a rational error bound $\varepsilon > 0$. Also assume that we know a rational, strictly feasible solution $\tilde{x}$, and a bound $R > 0$ for the coordinates of an optimal solution. Then the set K of feasible solutions is a closed, convex, bounded, full-dimensional set in $\mathbb{R}^n$. It is easy to compute a small ball around x_0 that is contained in K.

The key step is to design a separation oracle for K. Given a vector x, we need only check whether $x \in K$ and if not, find a separating hyperplane. Ignoring numerical problems, we can use the algorithm described in section 2.1 to check whether the matrix $Y = \sum_i x_i A_i - B$ is positive semidefinite. If it is, then $x \in K$. If not, the algorithm also returns a vector $v \in \mathbb{R}^m$ such that $v^\mathsf{T} Y v < 0$. Then $\sum_i x_i v^\mathsf{T} A_i v = v^\mathsf{T} B v$ is a separating hyperplane. (Because of numerical problems, the error bound in the definition of the weak separation oracle is needed.)

Thus using the ellipsoid method we can compute, in time polynomial in $\log(1/\varepsilon)$ and in the number of digits in the coefficients and in x_0, a feasible solution x such that the value of the objective function is at most $v_{\text{primal}} + \varepsilon$.

Unfortunately, the above argument gives an algorithm which is polynomial, but hopelessly slow, and practically useless. Still, the flexibility of the ellipsoid method makes it an inevitable tool in proving the *existence* (and not much more) of a polynomial time algorithm for many optimization problems.

Semidefinite programs can be solved in polynomial time and also *practically efficiently* by interior point methods [77, 1, 2]. The key to this method is the following property of the determinant of positive semidefinite matrices.

Lemma 3.8 *The function F defined by*

$$F(Y) = -\log \det\, (Y)$$

is convex and analytic in the interior of the semidefinite cone P_n*, and tends to* ∞ *at the boundary.*

The algorithm can be described very informally as follows. The feasible domain of our semidefinite optimization problem is of the form $K = \mathsf{P}_n \cap A$, where A is an affine subspace of symmetric matrices. We want to minimize a linear function $C \cdot X$ over $X \in K$. The good news is that K is convex. The bad news is that the minimum will be attained on the boundary of K, and this boundary can have a very complicated structure; it is neither smooth nor polyhedral. Therefore, neither gradient-type methods nor the methods of linear programming can be used to minimize $C \cdot X$.

The main idea of barrier methods is that instead of minimizing $C^{\mathsf{T}}X$, we minimize the function $F_C(X) = F(X)+\lambda C^{\mathsf{T}}X$ for some $\lambda > 0$. Since F_λ tends to infinity on the boundary of K, the minimum will be attained in the interior. Since F_λ is convex and analytic in the interior, the minimum can be very efficiently computed by a variety of numerical methods (conjugate gradient etc.)

Of course, the point we obtain this way is not what we want, but if λ is large it will be close. If we don't like it, we can increase λ and use the minimizing point for the old F_λ as the starting point for a new gradient type algorithm. (In practice, we can increase λ after each iteration of this gradient algorithm.)

One can show that (under some technical assumptions about the feasible domain) this algorithm gives an approximation of the optimum with relative error ε in time polynomial in $\log(1/\varepsilon)$ and the size of the presentation of the program. The proof of this depends on a further rather technical property of the determinant, called "self-concordance". We don't go into the details, but refer to the articles [2, 93, 94] and the book [76].

4 Obtaining semidefinite programs

How do we obtain semidefinite programs? It turns out that there are a number of considerations from which semidefinite programs, in particular semidefinite relaxations of combinatorial optimization problems arise. These don't always lead to different relaxations; in fact, the best known applications of semidefinite programming seem to be very robust in the sense that different methods for deriving their semidefinite relaxations yields the same, or almost the same, result. However, these different methods seem to have different heuristic power.

4.1 Unit distance graphs and orthogonal representations

We start with some semidefinite programs arising from geometric problems. A *unit distance representation* of a graph $G = (V, E)$ is a mapping $u : V \to \mathbb{R}^d$ for some $d \geq 1$ such that $|u_i - u_j| = 1$ for every $ij \in E$ (we allow that $|u_i - u_j| = 1$ for some $ij \in \overline{E}$). Figure 8 shows a 2-dimensional unit distance representation of the Petersen graph [31].

There are many questions one can ask about the existence of unit distance representations: what is the smallest dimension in which it exists? what is the smallest radius of a ball containing a unit distance representation of G (in any dimension)? In this paper, we are only concerned about the last question, which can be answered using semidefinite

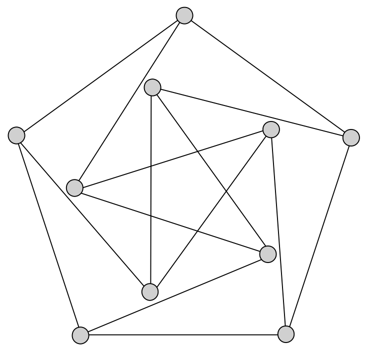

Figure 8: A unit distance representation of the Petersen graph.

programming (for a survey of other aspects of such geometric representations, see [73]). Considering the Gram matrix $A = (u_i^\top u_j)$, it is easy to obtain the following reduction to semidefinite programming:

Proposition 4.1 *A graph G has a unit distance representation in a ball of radius R (in some appropriately high dimension) if and only if there exists a positive semidefinite matrix A such that*

$$\begin{aligned} A_{ii} &\le R^2 \quad (i \in V) \\ A_{ii} - 2Aij + A_{jj} &= 1 \quad (ij \in E). \end{aligned}$$

In other words, the smallest radius R is the square root of the optimum value of the semidefinite program

$$\begin{array}{llcl} \text{minimize} & w & & \\ \text{subject to} & A & \succeq & 0 \\ & A_{ii} & \le & w \quad (i \in V) \\ & A_{ii} - 2Aij + A_{jj} & = & 1 \quad (ij \in E). \end{array}$$

The unit distance embedding of the Petersen graph in Figure 8 is not an optimal solution of this problem. Let us illustrate how semidefinite optimization can find the optimal embedding by determining this for the Petersen graph. In the formulation above, we have to find a 10×10 positive semidefinite matrix A satisfying the given linear constraints. For a given w, the set of feasible solutions is convex, and it is invariant under the automorphisms of the Petersen graph. Hence there is an optimum solution which is invariant under these automorphisms (in the sense that if we permute the rows and columns by the same automorphism of the Petersen graph, we get back the same matrix).

Now we know that the Petersen graph has a very rich automorphism group: not only can we transform every node into every other node, but also every edge into every other edge, and every nonadjacent pair of nodes into every other non-adjacent pair of nodes. A matrix invariant under these automorphisms has only 3 different entries: one number in the diagonal, another number in positions corresponding to edges, and a third number in positions corresponding to nonadjacent pairs of nodes. This means that this optimal matrix A can be written as

$$A = xP + yJ + zI,$$

where P is the adjacency matrix of the Petersen graph, J is the all-1 matrix, and I is the identity matrix. So we only have these 3 unknowns x, y and z to determine.

The linear conditions above are now easily translated into the variables x, y, z. But what to do with the condition that A is positive semidefinite? Luckily, the eigenvalues of A can also be expressed in terms of x, y, z. The eigenvalues of P are well known (and easy to compute): they are 3, 1 (5 times) and -2 (4 times). Here 3 is the degree, and it corresponds to the eigenvector $\mathbf{1} = (1, \ldots, 1)$. This is also an eigenvector of J (with eigenvalue 10), and so are the other eigenvectors of P, since they are orthogonal to $\mathbf{1}$, and so are in the nullspace of J. Thus the eigenvalues of $xP + yJ$ are $3x + 10y$, x, and $-2x$. Adding zI just shifts the spectrum by z, so the eigenvalues of A are $3x + 10y + z$, $x + z$, and $-2x + z$. Thus the positive semidefiniteness of A, together with the linear constraints above, gives the following linear program for x, y, z, w:

$$\begin{array}{llcl}
\text{minimize} & w & & \\
\text{subject to} & 3x + 10y + z & \geq & 0, \\
& x + z & \geq & 0, \\
& -2x + z & \geq & 0, \\
& y + z & \leq & w, \\
& 2z - 2x & = & 1.
\end{array}$$

It is easy to solve this: clearly the optimum solution will have $w = y + z$, and $y = (-3x - z)/10$. We can also substitute $x = z - 1/2$, which leaves us with a single variable. The solution is $x = -1/4$, $y = 1/20$, $z = 1/4$, and $w = 3/10$. Thus the smallest radius of a ball in which the Petersen graph has a unit distance representation is $\sqrt{3/10}$. The corresponding matrix A has rank 4, so this representation is in 4 dimension.

It would be difficult to draw a picture of this representation, but I can offer the following nice matrix, whose columns will realize this representation (the center of the smallest ball containing it is not at the origin!):

$$\begin{pmatrix}
1/2 & 1/2 & 1/2 & 1/2 & 0 & 0 & 0 & 0 & 0 & 0 \\
1/2 & 0 & 0 & 0 & 1/2 & 1/2 & 1/2 & 0 & 0 & 0 \\
0 & 1/2 & 0 & 0 & 1/2 & 0 & 0 & 1/2 & 1/2 & 0 \\
0 & 0 & 1/2 & 0 & 0 & 1/2 & 0 & 1/2 & 0 & 1/2 \\
0 & 0 & 0 & 1/2 & 0 & 0 & 1/2 & 0 & 1/2 & 1/2
\end{pmatrix} \tag{11}$$

(This matrix reflects the fact that the Petersen graph is the complement of the line-graph of K_5.)

It turns out that from a graph theoretic point of view, it is more interesting to modify the question and require that the nodes all lie on the surface of the sphere (in our example this happened automatically, due to the symmetries of the Petersen graph). In other words, we are interested in the smallest sphere (in any dimension) on which a given graph G can be drawn so that the euclidean distance between adjacent nodes is 1 (of course, we could talk here about any other given distance instead of 1, or spherical distance instead of euclidean, without essentially changing the problem). Again, by considering the Gram matrix $A = (u_i^\top u_j)$, we find that this smallest radius $t(G)$ is given by the square root of the optimum value of the following semidefinite program:

$$\begin{array}{llcll} \text{minimize} & z & & & \\ \text{subject to} & A & \succeq & 0 & \\ & A_{ii} & = & z & (i \in V) \\ & A_{ii} - 2Aij + A_{jj} & = & 1 & (ij \in E). \end{array} \tag{12}$$

Since $A = \mathrm{diag}(1/2, \dots, 1/2)$ is a solution, it follows that the optimal z satisfies $z \le 1/2$.

Another way of looking at this question is to add a further dimension. Think of a unit distance representation of the graph on the sphere with radius t as lying in a "horizontal" hyperplane. Choose the origin above the center of the sphere so that the vectors pointing to adjacent nodes of the graph are orthogonal (the distance of the origin to the hyperplane will be $\sqrt{(1/2) - z}$). It is worth scaling up by a factor of $\sqrt{2}$, so that the vectors pointing to the nodes of the graph become unit vectors. Such a system of vectors is called an *orthonormal representation* of the complementary graph $\overline{G}$ (the complementation is, of course, just a matter of convention). The matrix (11) above is an orthogonal representation of the complement of the Petersen graph, which is related to its unit distance representation by this construction, up to a change in coordinates.

In the introduction, we constructed an orthonormal representation of the pentagon graph (Figure 1). This is not the simplest case (in a sense, it is the smallest interesting orthogonal representation). Figure 9 below shows that it if we add a diagonal to the pentagon, then a much easier orthogonal representation in 2 dimensions can be constructed.

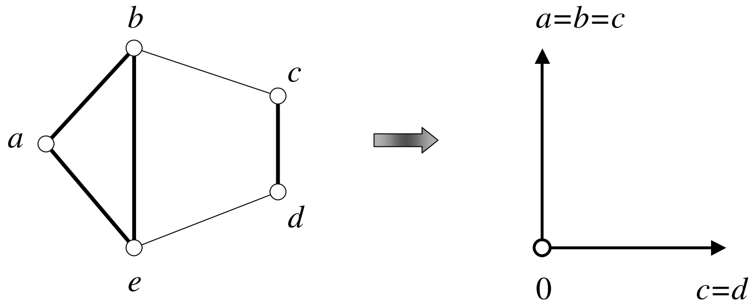

Figure 9: An (almost) trivial orthogonal representation

Orthogonal representations of graphs have several applications in graph theory. In particular, it turns out that the quantity $1/(1 - 2t(G)^2)$ is just $\vartheta(\overline{G})$ introduced before (for the complementary graph $\overline{G}$. We'll return to it in sections 5.1 and 6.1.

4.2 Discrete linear and quadratic programs

Consider a typical 0-1 optimization problem:

$$\text{maximize} \quad c^t x$$

$$\text{subject to} \qquad \begin{cases} Ax \le b \\ x \in \{0,1\}^n. \end{cases} \tag{13}$$

We get an equivalent problem if we replace the last constraint by the quadratic equation

$$x_i^2 = x_i \qquad (i = 1, \dots, n). \tag{14}$$

Once we allow quadratic equations, many things become much simpler. First, we can restrict ourselves to homogeneous quadratic equations, by introducing a new variable x_0, and setting it to 1. Thus (14) becomes

$$x_i^2 = x_0 x_i \qquad (i = 1, \dots, n). \tag{15}$$

Second, we don't need inequalities: we can just replace $F \ge 0$ by $F - x^2 = 0$, where x is a new variable. Third, we can often replace constraints by simpler and more powerful constraints. For example, for the stable set problem (section 2.3), we could replace the edge constraints by the quadratic equations

$$x_i x_j = 0 \qquad (ij \in E). \tag{16}$$

Trivially, the solutions of (14) and (16) are precisely the incidence vectors of stable sets. If we are interested in $\alpha(G)$, we can consider the objective function $\sum_{i=1}^n x_0 x_i$.

Unfortunately, this also shows that even the solvability of such a simple system of quadratic equations (together with a linear equation $\sum_i x_i = \alpha$) is NP-hard.

The trick to obtain a polynomially solvable relaxation of such problems is to think of the x_i as vectors in $\mathbb{R}^k$ (and multiplication as inner product). For $k = 1$, we get back the original 0-1 optimization problem. For $k = 2, 3 \dots$, we get various optimization problems with geometric flavor, which are usually not any easier than the original. For example, for the stable set problem we get the *vector relaxation*

$$\begin{aligned} \text{maximize} \qquad & \sum_{i \in V} v_0^\mathsf{T} v_i \\ \text{subject to} \qquad & v_i \in \mathbb{R}^k \\ & v_0^\mathsf{T} v_i = |v_i|^2 \quad (i \in V) && (17) \\ & v_i^\mathsf{T} v_j = 0 \quad (ij \in E). && (18) \end{aligned}$$

But if we take $k = n$, then we get a relaxation which is polynomial time solvable. Indeed, we can introduce new variables $Y_{ij} = v_i^\mathsf{T} v_j$ and then the constraints and the objective function become linear, and if in addition we impose the condition that $Y \succeq 0$, then we get a semidefinite optimization problem. If we solve this problem, and then write Y as a Gram matrix, we obtain an optimum solution of the vector relaxation.

The conditions on vector relaxations often have useful geometric content. For example, (17) (which is common to the vector relaxations of all 0-1 programs) can be written in the following two forms:

$$(v_0 - v_i)^\mathsf{T} v_i = 0; \qquad \left| v_i - \frac{1}{2} v_0 \right|^2 = \frac{1}{4}.$$

This says that the vectors v_i and $v_0 - v_i$ are orthogonal to each other, and all the points v_i lie on the sphere with radius 1/2 centered at $(1/2)v_0$. (18) says that the v_i form an orthogonal representation of the complement of G.

For discrete linear or quadratic programs with variables from $\{-1, 1\}$, (14) becomes even simpler:

$$x_i^2 = 1, \tag{19}$$

i.e., the vectors are unit vectors. In the case of the Maximum Cut Problem for a graph $G = (V, E)$, we can think of a 2-coloring as an assignment of 1's and -1's to the nodes, and the number of edges in the cut is

$$\sum_{ij \in E} \frac{1}{4}(x_i - x_j)^2.$$

The vector relaxation of this problem has the nice physical meaning given in the introductory example (energy-minimization).

One can add further constraints. For example, if the variables x_i are 0-1, then we have

$$(x_i - x_j)(x_i - x_k) \geq 0$$

for any three variables. We may add these inequalities as quadratic constraints, and then get a vector relaxation that satisfies, besides the other constraints, also

$$(v_i - v_j)^{\mathsf{T}}(v_i - v_k) \geq 0.$$

Geometrically, this means that every triangle spanned by the vectors v_i is acute; this property is sometimes useful to have.

A further geometric property that can be exploited in some cases is *symmetry.* Linear systems always have solutions invariant under the symmetries of the system, but quadratic systems, or discrete linear systems do not. For example, if G is a cycle, then the system (14)-(16) is invariant under rotation, but its only solution invariant under rotation is the trivial all-0 vector. One advantage of the semidefinite relaxation is that it restores symmetric solvability.

Assume that we start with a quadratic system such that both the constraint set and the objective function are invariant under some permutation group Γ acting on the variables (for example, it can be invariant under the cyclic shift of indices). It may be that no optimal solution of the quadratic system is invariant under these permutations: For example, no maximal stable set in a cycle is invariant under cyclic shifts. However, in a semidefinite program feasible solutions define convex sets in the space of matrices, and the objective function is linear. Hence by averaging, we can assert that there exists an optimum solution Y which itself is invariant under all permutations of the indices under which the semidefinite program is. In other words, the semidefinite relaxation of the quadratic system has an optimal solution $Y \succeq 0$, such that if $\gamma \in \Gamma$, then

$$Y_{\gamma(i),\gamma(j)} = Y_{ij}. \tag{20}$$

Now we go over to the vector relaxation: this is defined by $Y_{ij} = v_i^T v_j$, where $v_i \in \mathbb{R}^d$ for some $d \leq n$. We may assume that the v_i span $\mathbb{R}^d$. Let $\gamma \in \Gamma$. (20) says that

$v_{\gamma(i)}^T v_{\gamma(j)} = v_i^T v_j$. In other words, the permutation $v_i \mapsto v_{\gamma(i)}$ preserves the length of the u_i and all the angles between them, and hence there is an orthogonal matrix M_γ such that $u_{\gamma(i)} = M_\gamma u_i$. Since the u_i span the space, this matrix M_γ is uniquely determined, and so we get a representation of Γ in $\mathbb{R}^d$. The vector solution (v_i) is invariant under this representation.

4.3 Spectra of graphs

Let $G = (V, E)$ be a graph. We denote by $\overline{G} = (V, \overline{E})$ its complement and set $\Delta = \{ii : i \in V\}$. The adjacency matrix A_G of G is defined by

$$(A_G)_{ij} = \begin{cases} 1, & \text{if } ij \in E, \\ 0, & \text{if } ij \in \overline{E} \cup \Delta. \end{cases}$$

Let $\lambda_1 \geq \ldots \geq \lambda_n$ be the eigenvalues of A_G. It is well known and easy to show that if G is d-regular than $\lambda_1 = d$. Since the trace of A_G is 0, we have $\lambda_1 + \ldots + \lambda_n = 0$, and hence if $E \neq \emptyset$ then $\lambda_1 > 0$ but $\lambda_n < 0$.

There are many useful connections between the eigenvalues of a graph and its combinatorial properties. The first of these follows easily from interlacing eigenvalues.

Proposition 4.2 *The maximum size $\omega(G)$ of a clique in G is at most $\lambda_1 + 1$. This bound remains valid even if we replace the non-diagonal 0's in the adjacency matrix by arbitrary real numbers.*

The following bound on the chromatic number is due to Hoffman.

Proposition 4.3 *The chromatic number $\chi(G)$ of G is at least $1 - (\lambda_1/\lambda_n)$. This bound remains valid even if we replace the 1's in the adjacency matrix by arbitrary real numbers.*

The following bound on the maximum size of a cut is due to Delorme and Poljak [28, 29, 75, 81], and was the basis for the Goemans-Williamson algorithm discussed in the introduction.

Proposition 4.4 *The maximum size $\gamma(G)$ of a cut in G is at most $|E|/2 - (n/4)\lambda_n$. This bound remains valid even if we replace the diagonal 0's in the adjacency matrix by arbitrary real numbers.*

Observation: to determine the best choice of the "free" entries in 4.2, 4.3 and 4.4 takes a semidefinite program. Consider 4.2 for example: we fix the diagonal entries at 0, the entries corresponding to edges at 1, but are free to choose the entries corresponding to non-adjacent pairs of vertices (replacing the off-diagonal 1's in the adjacency matrix). We want to minimize the largest eigenvalue. This can be written as a semidefinite program:

$$\begin{array}{lrcll} \text{minimize} & t & & & \\ \text{subject to} & tI - X & \succeq & 0, & \\ & X_{ii} & = & 0 & (\forall i \in V), \\ & X_{ij} & = & 1 & (\forall ij \in E). \end{array}$$

It turns out that the semidefinite program constructed for 4.3 is just the dual of this, and their common optimum value is the parameter $\vartheta(G)$ introduced before. The program for 4.4 gives the approximation used by Goemans and Williamson (for the case when all weights are 1, from which it is easily extended). See [50] for a similar method to obtain an improved bound on the mixing rate of random walks.

4.4 Engineering applications

Semidefinite optimization has many applications in stability problems of dynamical systems and optimal control. Since this is not in the main line of these lecture notes, we only illustrate this area by a simple example; see chapter 14 of [98] for a detailed survey.

Consider a "system" described by the differential equation

$$\frac{dx}{dt} = A(t)x(t), \tag{21}$$

where $x \in \mathbb{R}^n$ is a vector describing the state of the system, and $A(t)$ is an $n \times n$ matrix, about which we only know that it is a linear combination of m given matrices $A_1, \dots, A_m$ with nonnegative coefficients (an example of this situation is when we know the signs of the matrix entries). Is the zero solution $x(t) \equiv 0$ asymptotically stable, i.e., is it true that for every initial value $x(0) = x_0$, we have $x(t) \to 0$ as $t \to \infty$?

Suppose first that $A(t) = A$ is a constant matrix, and also suppose that we know from the structure of the problem that it is symmetric. Then the basic theory of differential equations tells us that the zero solution is asymptotically stable if and only if A is negative definite.

But semidefinite optimization can be used even if $A(t)$ can depend on t, and is not necessarily symmetric, at least to establish a sufficient condition for asymptotic stability. We look for a *quadratic Lyapunov function* $x^{\mathsf{T}}Px$, where P is a positive definite $n \times n$ matrix, such that

$$\frac{d}{dt} x(t)^T P x(t) < 0 \tag{22}$$

for every non-zero solution of the differential equation. If we find such a matrix P, then Lyapunov's theorem implies that the trivial solution is asymptotically stable.

Now the left hand side of (22) can be written as

$$\frac{d}{dt} x(t)^T P x(t) = \dot{x}^{\mathsf{T}} P x + x^{\mathsf{T}} P \dot{x} = x^{\mathsf{T}}(A^{\mathsf{T}}P + PA)x.$$

Thus (22) holds for every solution and every t if and only if $A^{\mathsf{T}}P + PA$ (which is a symmetric matrix) is negative semidefinite. We don't explicitly know $A(t)$, but we do know that it is a linear combination of $A_1, \dots, A_m$; so it suffices we require that the matrices $A_i^{\mathsf{T}}P + PA_i$, $i = 1, ..., m$ are negative semidefinite.

To sum up, we see that a sufficient condition for the asymptotic stability of the zero solution of (21) is that the semidefinite system

$$\begin{aligned} P &\succ 0, \\ -A^{\mathsf{T}}P - PA &\succ 0 \quad (i = 1, \dots, m) \end{aligned}$$

has a solution in P.

5 Semidefinite programming in proofs

5.1 More on stable sets and the Shannon capacity

An *orthogonal representation* of a graph $G = (V, E)$ is a mapping (labeling) $u: V \to \mathbb{R}^d$ for some d such that $u_i^\top u_j = 0$ for all $ij \in \overline{E}$. An *orthonormal representation* is an orthogonal representation with $|u_i| = 1$ for all i. The *angle* of an orthonormal representation is the smallest half-angle of a rotational cone containing the representing vectors.

Proposition 5.1 *The minimum angle ϕ of any orthogonal representation of G is given by $\cos^2\phi = 1/\vartheta(G)$.*

In what follows we collect some properties of ϑ, mostly from [64] (see also [57] for a survey).

We start with a formula that expresses $\vartheta(G)$ as a maximum over orthogonal representations of the *complementary* graph. Let the *leaning* of an orthonormal representation of G be defined as $\sum_{i\in V}(e_1^\top u_i)^2$.

Proposition 5.2 *The maximum leaning of an orthonormal representation of G is $\vartheta(\overline{G})$.*

The "umbrella" construction given in the introduction shows, by Proposition 5.1, that $\vartheta(C_5) \le \sqrt{5}$, and by Proposition 5.2, that $\vartheta(\overline{C_5}) \ge \sqrt{5}$. Hence $\vartheta(C_5) = \sqrt{5}$.

Proposition 5.2 is a "duality" result, which is in fact a consequence of the Duality Theorem of semidefinite programs (Theorem 3.4). To see the connection, let us give a "semidefinite" formulation of ϑ. This formulation is by no means unique; in fact, several others come up in these lecture notes.

Proposition 5.3 *$\vartheta(G)$ is the optimum of the following semidefinite program:*

$$\begin{array}{lrcl} \text{minimize} & t & & \\ \text{subject to} & Y & \succeq & 0 \\ & Y_{ij} & = & -1 \quad (\forall\ ij \in E(\overline{G})) \\ & Y_{ii} & = & t-1 \end{array} \tag{23}$$

It is also the optimum of the dual program

$$\begin{array}{lrcl} \text{maximize} & \sum_{i\in V}\sum_{j\in V} Z_{ij} & & \\ \text{subject to} & Z & \succeq & 0 \\ & Z_{ij} & = & 0 \quad (\forall\ ij \in E(G)) \\ & \mathrm{tr}(Z) & = & 1 \end{array} \tag{24}$$

Any stable set S provides a feasible solution of (24), by choosing $Z_{ij} = 1/|S|$ if $i, j \in S$ and 0 otherwise. Similarly, any k-coloring of $\overline{G}$ provides a feasible solution of (23), by choosing $Y_{ij} = -1$ if i and j have different colors, $Y_{ii} = k-1$ and $Y_{ij} = 0$ otherwise. These explicit solutions imply the following.

Theorem 5.4 [Sandwich Theorem] *For every graph G,*

$$\omega(G) \le \vartheta(\overline{G}) \le \chi(G).$$

The *fractional chromatic number* $\chi^*(G)$ is defined as the least t for which there exists a family $(A_j : \ j = 1, \ldots, p)$ of stable sets in G, and nonnegative weights $(\tau_j : \ j = 1, \ldots, p)$ such that $\sum\{\tau_j : \ A_j \ni i\} \geq 1$ for all $i \in V$ and $\sum_j \tau_j = t$. Note that the definition χ^* can be considered as a linear program. By linear programming duality, $\chi^*(G)$ is equal to the largest s for which there exist weights $(\sigma_i : \ i \in V)$ such that $\sum_{i \in A} \sigma_i \leq 1$ for every stable set A and $\sum_i \sigma_i = s$.

Clearly $\omega(G) \leq \chi^*(G) \leq \chi(G)$.

Proposition 5.5 $\vartheta(G) \leq \chi^*(\overline{G})$.

Returning to orthogonal representations, it is easy to see that not only the angle, but also the dimension of the representation yields an upper bound on $\alpha(G)$. This is, however, not better that ϑ:

Proposition 5.6 *Suppose that G has an orthonormal representation in dimension d. Then $\vartheta(G) \leq d$.*

On the other hand, if we consider orthogonal representations over fields of finite characteristic, the dimension may be a better bound than ϑ [44, 6]. This, however, goes outside the ideas of semidefinite optimization.

To relate ϑ to the Shannon capacity of a graph, the following is the key observation:

Proposition 5.7 *For any two graphs,*

$$\vartheta(G \cdot H) = \vartheta(G)\vartheta(H)$$

and

$$\vartheta(\overline{G \cdot H}) = \vartheta(\overline{G})\vartheta(\overline{H}).$$

It is now easy to generalize the bound for the Shannon capacity of the pentagon, given in the introduction, to arbitrary graphs.

Corollary 5.8 *For every graph,*

$$\Theta(G) \leq \vartheta(G).$$

Does equality hold here? Examples by Haemers [44], and more recent much sharper examples by Alon [6] show that the answer is negative in general. But we can derive at least one interesting class of examples from the general results below.

Proposition 5.9 *For every graph G,*

$$\vartheta(G)\vartheta(\overline{G}) \geq n.$$

If G has a vertex-transitive automorphism group, then equality holds.

Corollary 5.10 *If G is a self-complementary graph on n nodes with a node-transitive automorphism group, then*

$$\Theta(G) = \vartheta(G) = \sqrt{n}.$$

An example to which this corollary applies is the *Paley graph*: for a prime $p \equiv 1 \pmod 4$, we take the $\{0, 1, \dots, p-1\}$ as vertices, and connect two of them iff their difference is a quadratic residue. Thus we get an infinite family for which the Shannon capacity is non-trivial (i.e., $\Theta > \alpha$), and can be determined exactly.

The Paley graphs are quite similar to random graphs, and indeed, for random graphs ϑ behaves similarly:

Theorem 5.11 (Juhász [49]) *If G is a random graph on n nodes then $\sqrt{n} < \vartheta(G) < 2\sqrt{n}$ with probability $1 - o(1)$.*

It is not known how large the Shannon capacity of a random graph is.

We conclude this section by using semidefinite optimization to add further constraints to the stable set polytope (continuing the treatment in section 2.3). For every orthonormal representation $(v_i : \ i \in V)$ of $\overline{G}$, we consider the linear constraint

$$\sum_{i \in V} (e_1^\mathsf{T} v_i)^2 x_i \le 1. \tag{25}$$

It is easy to see that these inequalities are valid for STAB(G); we call them *orthogonality constraints*. The solution set of non-negativity and orthogonality constraints is denoted by TSTAB(G). It is clear that TSTAB is a closed, convex set. The incidence vector of any stable set A satisfies (25). Indeed, it then says that

$$\sum_{i \in A} (e_1^\mathsf{T} v_i)^2 \le 1.$$

Since the v_i $(i \in A)$ are mutually orthogonal, the left hand side is just the squared length projection of e_1 onto the subspace spanned by these e_i, and the length of this projection is at most the length of e_1, which is 1.

Furthermore, every clique constraint is an orthogonality constraint. Indeed,

$$\sum_{i \in B} x_i \le 1$$

is the constraint derived from the orthogonal representation

$$i \mapsto \begin{cases} e_1, & \text{if } i \in A, \\ e_i, & \text{if } i \notin A. \end{cases}$$

Hence we have

$$\text{STAB}(G) \subseteq \text{TSTAB}(G) \subseteq \text{QSTAB}(G)$$

for every graph G.

There is a dual characterization of TSTAB [42], which can be derived from semidefinite duality. For every orthonormal representation $(u_i : \ i \in V)$, consider the vector $x[u] = (e_1^\mathsf{T} u_i)^2 : \ i \in V) \in \mathbb{R}^V$.

Theorem 5.12 TSTAB$(G) = \{x[u] : \ u$ is an orthonormal representation of $G\}$.

Not every orthogonality constraint is a clique constraint; in fact, the number of essential orthogonality constraints is infinite in general:

Theorem 5.13 TSTAB(G) *is polyhedral if and only if the graph is perfect. In this case* TSTAB = STAB = QSTAB.

While TSTAB is a rather complicated set, in many respects it behaves much better than, say, STAB. For example, it has a very nice connection with graph complementation:

Theorem 5.14 TSTAB($\overline{G}$) *is the antiblocker of* TSTAB(G).

Maximizing a linear function over STAB(G) or QSTAB(G) is NP-hard; but, surprisingly, TSTAB behaves much better:

Theorem 5.15 *Every linear objective function can be maximized over* TSTAB(G) *(with arbitrarily small error) in polynomial time.*

The maximum of $\sum_i x_i$ over TSTAB(G) is the familiar function $\vartheta(G)$.

5.2 Discrepancy and number theory

Let $\mathcal{F}$ be a family of subsets of $\{0, 1, \ldots, n-1\}$. We want to find a sequence $x = (x_0, x_1, \ldots, x_{n-1})$ of ± 1's so that each member of $\mathcal{F}$ contains about as many 1's as -1's. More exactly, we define the *discrepancy of the sequence* x by

$$\max_{A \in \mathcal{F}} \left| \sum_{i \in A} x_i \right|,$$

and the *discrepancy of the family* $\mathcal{F}$ by

$$\Delta(\mathcal{F}) = \min_{x \in \{-1,1\}^n} \max_{A \in \mathcal{F}} \left| \sum_{i \in A} x_i \right|.$$

We can also consider the "average discrepancy" in various versions. For our purposes, we only need the ℓ_2-*discrepancy*

$$\Delta_2(\mathcal{F}) = \min_{x \in \{-1,1\}^n} \frac{1}{|\mathcal{F}|} \sum_{A \in \mathcal{F}} \left(\sum_{i \in A} x_i \right)^2.$$

It is clear that $\Delta_2 \le \Delta^2$. (We refer to [17] and [18] for an exposition of combinatorial discrepancy theory.)

Clearly, $\Delta(\mathcal{F})$ can be thought of as the optimum of a linear program in $\{-1, 1\}$-variables:

$$\begin{array}{lrcl} \text{minimize} & t & & \\ \text{subject to} & -t & \le & \sum_{i \in A} x_i \le t \\ & x_i \in \{-1, 1\}, & & \end{array} \tag{26}$$

while Δ_2 is optimum of a quadratic function in $\{-1,1\}$-variables (but otherwise unconstrained). So both quantities have natural semidefinite relaxations. We only formulate the second:

$$\begin{array}{llcl} \text{minimize} & \frac{1}{|\mathcal{F}|}\sum_{A\in\mathcal{F}}\sum_{i\in A}\sum_{j\in A} Y_{ij} & & \\ \text{subject to} & Y & \succeq & 0, \\ & Y_{ii} & = & 1 \quad (\forall\, i \in V). \end{array} \tag{27}$$

We show how to use the semidefinite relaxation to estimate $\Delta(\mathcal{F})$ in the case when $\mathcal{F}$ is the family of arithmetic progressions in $\{0,1,\dots,n-1\}$ [68]. One way of looking at this particular question is to think of the x_i in the definition of discrepancy as the output of a pseudorandom number generator, and of the discrepancy, as a randomness test (a quantitative version of von Mises' test). If the x_i are truly random, we expect this discrepancy to be about $n^{1/2}$. Most "bad" sequences one encounters fail by producing a larger discrepancy. Can a sequence fail by producing a discrepancy that is too small?

The theorem of Roth [85] below shows that the discrepancy $\Delta(\mathcal{F})$ cannot be smaller than $\Omega(n^{1/4})$; this allows sequences to have substantially smaller discrepancy than a random sequence. One might expect that the lower bound in the theorem can be strengthened to about $\Omega(n^{1/2})$ (so that the random sequences would have, at least approximately, the smallest discrepancy), but it was shown by Beck [16] that Roth's estimate is sharp up to a logarithmic factor. Recently, even this logarithmic factor was removed by Matoušek and Spencer [74].

Theorem 5.16 *For every sequence $(x_0,\dots,x_{n-1})$, $x_i \in \{-1,1\}$, there is an arithmetic progression $A \subseteq \{0,\dots,n-1\}$ such that*

$$\left|\sum_{i\in A} x_i\right| > \frac{1}{14} n^{1/4}.$$

All proofs of this theorem establish more: one has such an arithmetic progression A with difference at most $8k$ and length exactly k, where $k = \lfloor\sqrt{n/8}\rfloor$. We consider arithmetic progressions modulo n, *i.e.*, we let them wrap around. (Of course, in this case it may happen that the progression with the large discrepancy is wrapped; but since $(k-1)(8k) < n$, it wraps over n at most once, and so it is the union of two unwrapped arithmetic progressions, one of which has discrepancy at least half the original.) Let $\mathcal{H}$ denote the family of such arithmetic progressions. Clearly $|\mathcal{H}| = 8kn$.

Following Roth, we prove the stronger result that the ℓ_2-discrepancy of arithmetic progressions in $\mathcal{H}$ is at least $(1/49)n^{1/2}$; even stronger, we prove that the optimum of its semidefinite relaxation is large: the minimum of

$$\frac{1}{|\mathcal{H}|}\sum_{A\in\mathcal{H}}\sum_{i\in A}\sum_{j\in A} Y_{ij} \tag{28}$$

subject to

$$Y \succeq 0, \tag{29}$$

$$Y_{ii} = 1 \quad (1 \le i \le n) \tag{30}$$

is at least $(1/49)n^{1/2}$.

The next step is to notice that both (30) and (29) are invariant under the cyclic shift of indices. Hence by our discussions in section 4.2, we have an optimal vector solution $(u_0, \ldots, u_n)$, and an orthogonal matrix M such that $M^n = I$ and $u_i = M^i u_0$.

Elementary group representation theory tells us that the space decomposes into the direct sum of 1- and 2-dimensional subspaces invariant under M. In other words, if we choose a basis appropriately, M has a block-diagonal form

$$M = \begin{pmatrix} M_1 & 0 & \ldots & 0 \\ 0 & M_2 & \ldots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \ldots & M_d \end{pmatrix}$$

where each M_t is a 1×1 or 2×2 real matrix of order n.

We show that the statement is true if M has only one block (thus $d = 1$ or 2). The general case then follows easily by adding up the lower bounds on the objective function for all diagonal blocks. We treat the case $d = 2$; the case $d = 1$ is trivial.

The matrix M defines a rotation in the plane with an angle $2\pi a/n$ for some $1 \le a \le n$. By Dirichlet's Theorem, there are integers $1 \le q \le 8k$ and p such that $|q(a/n) - p| < 1/(8k)$. This implies that for every arithmetic progression A of difference q and length k, the vectors $M^j u_0$ $(j \in A)$ point in almost the same direction: the maximum angle between them is less than $(k-1)(2\pi/(8k)) < \pi/4$. Hence

$$\left| \sum_{j \in A} M^j u_0 \right|^2 > \frac{k^2}{2}.$$

Since there are n arithmetic progressions in $\mathcal{H}$ with this difference, we get

$$\frac{1}{8kn} \sum_{A \in \mathcal{H}} \left| \sum_{j \in A} M^j u_0 \right|^2 > \frac{1}{8kn} \frac{k^2 n}{2} = \frac{k}{16} > \frac{n^{1/2}}{49},$$

as claimed.

6 Semidefinite programming in approximation algorithms

The algorithm of Goemans and Williamson, discussed in the introduction, was a breakthrough which showed that semidefinite optimization can lead to approximation algorithms with very good approximation ratio. Since then, many other applications have been developed; a couple of these are discussed below.

6.1 Stable sets, cliques, and chromatic number

The Sandwich Theorem 5.4 implies that $\vartheta(\overline{G})$ can be considered as an approximation of the clique size $\omega(G)$, which is at least as good as the natural upper bound $\chi(G)$. Note that both quantities $\omega(G)$ and $\chi(G)$ are NP-hard, but $\vartheta(\overline{G})$, which is "sandwiched" between them, is polynomial time computable.

The most important algorithmic consequence of theorem 5.4 is that *for perfect graphs,* $\omega(G) = \chi(G)$ *is polynomial time computable* [41]. Of course, by complementation it follows that $\alpha(G)$ is also polynomial time computable. It is not hard to see how to use this algorithm to compute a maximum stable set and (with more work) an optimum coloring. The surprising fact is that there is no algorithm known to find a maximum stable set in a perfect graph without the use of semidefinite optimization. (For another application of this result to complexity theory, see [90].)

How good an approximation does ϑ provide for α? Unfortunately, it can be quite bad. First, consider the case when α is very small. Koniagin [55] constructed a graph that has $\alpha(G) = 2$ and $\vartheta(G) = \Omega(n^{1/3})$. This is the largest $\vartheta(G)$ can be; in fact, Alon and Kahale [8], improving results of Kashin and Koniagin [54], proved that if $\alpha(G) \leq k$ then $\vartheta(G) < Cn^{(k-1)/(k+1)}$, for some absolute constant C.

Once α is unbounded, very little is true. Feige [32] showed that there are graphs for which $\alpha(G) = n^{o(1)}$ and $\vartheta(G) = n^{1-o(1)}$; in other words, ϑ/α can be larger than $n^{1-\varepsilon}$ for every $\varepsilon > 0$. (The existence of such graphs also follows from the results of Håstad [46] showing that it is NP-hard to determine $\alpha(G)$ with a relative error less than $n^{1-\varepsilon}$, where $n = |V|$.) By results of Szegedy [89], this also implies that $\vartheta(\overline{G})$ does not approximate the chromatic number within a factor of $n^{1-\varepsilon}$.

Let us consider the other end of the scale, when $\vartheta(\overline{G})$ is small. Suppose first that $\vartheta(\overline{G}) = 2$, then $\vartheta(\overline{G}) = 2$. Then it is not hard to see that G is bipartite, and hence perfect, and hence $\vartheta(G) = \alpha(G)$.

For the case when $\vartheta(\overline{G})$ is larger than 2 but bounded, the following (much weaker) positive result was proved by Karger, Motwani and Sudan [51]:

Theorem 6.1 *Let* $k = \lceil \vartheta(\overline{G}) \rceil$*, then* $\alpha(G) \geq (1/2)n^{3/(k+1)}/\sqrt{\ln n}$*. Furthermore, a stable set of this size can be found in randomized polynomial time.*

Note that we have $\vartheta(G) \geq n/k$ by Proposition 5.9. It is not known how large a stable set follows from the assumption $\vartheta(G) \geq n/k$.

Let us sketch the algorithm. If $k = 2$ then a stronger bound holds, as discussed above, so suppose that $k > 2$.

We first treat the case when the maximum degree of the graph is $\Delta > n^{k/(k+1)}$. Let G' be the subgraph induced by the neighbors of a node with maximum degree. It is easy to see that $\vartheta(G') \leq k - 1$, and so (by induction on k) we can find in G' a stable set of size at least $\Delta^{3/k}/\sqrt{\ln \Delta} \geq n^{3/(k+1)}/\sqrt{\ln n}$.

So suppose that $\Delta \leq n^{k/(k+1)}$. Compute the optimum solution of (12) for the complementary graph $\overline{G}$, and the corresponding vector representation. Thus we get unit vectors $u_i \in \mathbb{R}^d$ such that for every edge $ij \in E$, we have $u_i^{\mathsf{T}} u_j = -1/(k-1)$.

Next, we take a random vector $w \in \mathbb{R}^d$ from the standard normal distribution in $\mathbb{R}^d$, and consider the set S of nodes i such that $w^{\mathsf{T}} u_i \geq c$, where $c = \sqrt{2(\ln n)(k-2)/k}$. The probability that a given node belongs to S is

$$\frac{1}{\sqrt{\pi}} \int_c^{\infty} e^{-t^2/2}\, dt \geq n^{-(k-2)/(k+1)}/\sqrt{\ln n},$$

and hence the expected size of S is at least $n^{3/(k+1)}/\sqrt{\ln n})$. On the other hand, the probability that both endpoints u_i and u_j of an edge belong to S can be estimated as follows:

$$\mathsf{P}(w^\mathsf{T} u_i \ge c,\ w^\mathsf{T} u_j \ge c) \le \mathsf{P}(w^\mathsf{T}(u_i + u_j) \ge 2c).$$

The conditions on the vector solution imply that $|u_i + u_j| = \sqrt{2(k-2)/(k-1)}$, and using this a more elaborate computation shows that the expected number of edges spanned by S is less than $|S|/2$. Hence we can delete at most half of the nodes of S and get a stable set of the desired size.

The previous algorithm has an important application to a coloring problem. Suppose that somebody gives a graph and guarantees that the graph is 3-colorable, without telling us its 3-coloring. Can we find this 3-coloring? (This may sound artificial, but this kind of situation does arise in cryptography and other data security applications; one can think of the hidden 3-coloring as a "watermark" that can be verified if we know where to look.)

It is easy to argue that knowing that the graph is 3-colorable does not help: it is still NP-hard to find the 3-coloration. But suppose that we would be satisfied with finding a 4-coloration, or 5-coloration, or $(\log n)$-coloration; is this easier? It is known that to find a 4-coloration is still NP-hard, but little is known above this. Improving earlier results, Karger, Motwani and Sudan [51] gave a polynomial time algorithm that, given a 3-colorable graph, computes a coloring with $O(n^{1/4}(\ln n)^{3/2})$ colors. More recently, this was improved by Blum and Karger [20] to $O(n^{3/14})$.

The algorithm of Karger, Motwani and Sudan starts with computing $\vartheta(\overline{G})$, which is at most 3 by Theorem 5.4. Using Theorem 6.1, they find a stable set of size $\Omega(n^{3/4}/\sqrt{\ln n})$. Deleting this set from G and iterating, they get a coloring of G with $O(n^{1/4}(\ln n)^{3/2})$ colors.

6.2 Satisfiability

One of the most fundamental problems in computer science is satisfiability. Let $x_1, \dots, x_n$ be Boolean variables. A *literal* is a variable x_i or the negation of a variable $\overline{x_i}$. A *clause* is a disjunction (OR) of literals; a *conjunctive normal form* is a conjunction (AND) of clauses. In standard logics notation, the following formula is an example of a conjunctive normal form:

$$(x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (x_4 \vee \overline{x_5}) \wedge (x_2 \vee \overline{x_3} \vee x_5).$$

The Satisfiability Problem (SAT) is the problem of deciding whether there is an assignment of values 0 or 1 to the variables that satisfies a given conjunctive normal form. The restricted case when we assume that each clause in the input has at most k literals is called k-SAT (the example above is an instance of 3-SAT). k-SAT is polynomial time solvable by a rather easy algorithm if $k = 2$, but NP-hard if $k > 2$.

Suppose that the given conjunctive normal form is not satisfiable; then we may want to find an assignment that satisfies as many clauses as possible; this optimization problem is called the MAX-SAT problem (we could assign weights to the clauses, and try to maximize the total weight of satisfied clauses; but we keep our discussion simple by assuming that all clauses are equally valuable). The restricted case of MAX-k-SAT is defined in the natural way. MAX-k-SAT is NP-hard already when $k = 2$; indeed, it is easy to see that MAX CUT is a special case.

Can we extend the semidefinite programming method so successful for MAX CUT to obtain good approximation algorithms for MAX-k-SAT? This idea was exploited already by Goemans and Williamson [38], who showed how to obtain for MAX-2-SAT the same approximation ratio .878 as for the MAX CUT problem; this was improved by Feige and Goemans [34] to .931.

We do not survey all the developments for various versions of the Satisfiability Problem, only the case of MAX-3-SAT. An important special case will be *exact MAX-3-SAT*, when all clauses contain exactly 3 literals.

In the negative direction, Håstad [45] proved that for the exact MAX-3-SAT problem no polynomial time approximation algorithm can have an approximation ratio better than 7/8 (unless P=NP). This approximation ratio is easy to achieve, since if we randomly assign values to the variables, we can expect to satisfy 7/8-th of all clauses.

Can this optimal approximation ratio be achieved in the more general case of MAX-3-SAT (when the clauses may contain 1, 2 or 3 literals)? Of course, Håstad's negative result remains valid. Using semidefinite optimization, Karloff and Zwick [53] (cf. also [99]) showed that this bound can be attained:

Theorem 6.2 *There is a polynomial time approximation algorithm for MAX-3-SAT with an approximation ratio of* 7/8.

Let us sketch this algorithm. First, we give a quadratic programming formulation. Let $x_1, \dots, x_n$ be the original variables, where we consider TRUE=1 and FALSE=0. Let $x_{n+i} = 1 - x_i$ $(i = n+1, \dots, 2n)$ be their negations. Let x_0 be a further variable needed for homogenization, which is set to $x_0 = 1$. We also introduce a variable $z_C \in 0, 1$ for the logical value of each clause C. Then we can relate z_C algebraically to the x_i as follows. For a clause $C = x_i$, we have $z_C = x_i$. For a clause $C = x_i \vee x_j$, we have $z_C = x_i + x_j - x_i x_j$. So far, this is all linear or quadratic, but clauses with 3 literals are a bit more difficult. If $C = x_i \vee x_j \vee x_k$, then clearly

$$z_C = x_i + x_j + x_k - x_i x_j - x_i x_k - x_j x_k + x_i x_j x_k.$$

unfortunately, this is cubic. We could get an upper bound on z_C if we omitted the last term, but as we will see, we need a lower bound. So we delete the cubic term and one of the quadratic terms; then we do get a lower bound. But which quadratic term should we delete? The trick is to create three inequalities, deleting one at a time:

$$\begin{aligned}
z_C &\geq x_i + x_j + x_k - x_i x_j - x_i x_k \\
z_C &\geq x_i + x_j + x_k - x_i x_j - x_j x_k \\
z_C &\geq x_i + x_j + x_k - x_i x_k - x_j x_k
\end{aligned}$$

Writing these expressions in a homogeneous form, we get the following optimization problem:

$$\begin{aligned}
x_0 x_i + x_0 x_j + x_0 x_k - x_i x_j - x_i x_k \geq z_C & \qquad \forall \text{ clause } C = x_i \vee x_j \vee x_k \\
x_0 x_i + x_0 x_j - x_i x_j = z_C & \qquad \forall \text{ clause } C = x_i \vee x_j \\
x_i = z_C & \qquad \forall \text{ clause } C = x_i \\
x_{n+i} = x_0 - x_i & \qquad \forall\, 1 \leq i \leq n, \\
x_i, z_C \in \{0, 1\}. &
\end{aligned} \tag{31}$$

It is easy to see that every assignment of the variables x_i and the values z_C determined by them give a solution of this system, and vice versa. Thus the value M of the MAX-3-SAT problem is the maximum of $\sum_C z_C$, subject to (31).

Now we consider the semidefinite relaxation where we replace the x_i by unit vectors; the variables z_C are relaxed to real values satisfying $0 \le z_C \le 1$. Using semidefinite programming, this can be solved in polynomial time (with an arbitrarily small error, which causes some complications to be ignored here).

Next, similarly as in the Goemans–Williamson algorithm, we take a random hyperplane H through the point $(1/2)v_0$, and set $x_i = 1$ if x_i is separated from 0 by H, and $x_i = 0$ otherwise. A clause with at most 2 variables will be satisfied with probability at least $.878 z_C > (7/8)z_C$ (which follows similarly as in the case of the Maximum Cut problem). A clause with 3 variables will be satisfied with probability at least $(7/8)z_C$ (this is quite a bit more difficult to show). Hence the expected number of clauses that are satisfied is at least

$$\sum_C \frac{7}{8} z_C = \frac{7}{8} M.$$

7 Constraint generation and quadratic inequalities

7.1 Example: the stable set polytope again

Recall the stable set polytope of a graph $G = (V, E)$ is the convex hull of integer solutions of the following system of linear inequalities:

$$x_i \ge 0 \qquad (\forall\, i \in V) \tag{32}$$

$$x_i + x_j \le 1 \qquad (\forall\, ij \in E) \tag{33}$$

Without the integrality condition, however, this system describes the larger polytope FSTAB. We discussed above how to add new faces to get a sufficiently large set of inequalities for certain classes of graphs. The additional constraints were obtained by *ad hoc* combinatorial considerations. We show now that many of them (in fact, all those mentioned above) can also be derived by algebraic arguments ([71, 72]; see also [67]).

The trick is to go quadratic. As we have seen, the fact that the variables are 0-1 valued implies that for every node i,

$$x_i^2 = x_i, \tag{34}$$

and the fact that x is the incidence vector of a stable set can be expressed as

$$x_i x_j = 0 \qquad (ij \in E). \tag{35}$$

Now we can start deriving inequalities, using only (34) and (35). We have

$$x_i = x_i^2 \ge 0,$$

and

$$1 - x_i - x_j = 1 - x_i - x_j + x_i x_j = (1 - x_i)(1 - x_j) \ge 0, \tag{36}$$

so (32) and (33) follow. These are rather trivial, so let us consider the odd hole constraint associated with a pentagon $(1,2,3,4,5)$. Then we have

$$\begin{aligned} 1 - x_1 - x_2 - x_3 + x_1x_3 &= 1 - x_1 - x_2 - x_3 + x_1x_2 + x_1x_3 \\ &= (1 - x_1)(1 - x_2 - x_3) \geq 0, \end{aligned}$$

and similarly

$$1 - x_1 - x_4 - x_5 + x_1x_4 \geq 0.$$

Furthermore,

$$x_1 - x_1x_3 - x_1x_4 = x_1(1 - x_3 - x_4) \geq 0$$

Summing these inequalities, we get the odd hole constraint

$$2 - x_1 - x_2 - x_3 - x_4 - x_5 \geq 0. \tag{37}$$

One obtains all odd hole constraints in a similar way.

We can also derive the clique constraints. Assume that nodes 1,2,3,4,5 induce a complete 5-graph. Then

$$\begin{aligned} 0 &\leq (1 - x_1 - x_2 - x_3 - x_4 - x_5)^2 = 1 + \sum_{i=1}^{5} x_i^2 - 2\sum_{i=1}^{5} x_i + 2\sum_{i \neq j} x_ix_j \\ &= 1 - x_1 - x_2 - x_3 - x_4 - x_5, \end{aligned}$$

by (34) and (35). All clique constraints, and in fact all orthogonality constraints can be derived similarly. Odd antihole constraints can be derived from the clique constraints in a way similar to the derivation of the odd hole constraints.

7.2 Strong insolvability of quadratic equations

We describe the procedures behind the computations in the previous section in a general context. We consider quadratic inequalities in n real variables $x_1, \dots, x_n$. Unfortunately, for quadratic inequalities there is no full analogue of the Farkas Lemma or of the efficient algorithms of linear programming. In fact, the system consisting of the quadratic equations (14) and (16), and a single linear equation $\sum_i x_i = k$ has a solution if and only if $\alpha(G) \geq k$. This reduction shows:

Proposition 7.1 *It is NP-hard to decide whether a system of quadratic inequalities has a real solution.*

However, using a semidefiniteness test for matrices, at least the case of a single inequality is solvable:

Proposition 7.2 *We can decide in polynomial time whether a single quadratic inequality is solvable. In fact, the quadratic polynomial*

$$q(x) = x^{\mathsf{T}}Ax + b^{\mathsf{T}}x + c$$

(where A is an $n \times n$ symmetric matrix, $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$) is everywhere positive if and only if

(a) $A \succeq 0$,

(b) $b = Ah$ *for some* $h \in \mathbb{R}^n$, *and*

(c) *for this* h, $h^\mathsf{T} b < 4c$.

These conditions are easy to verify.

A system of quadratic inequalities is *strongly unsolvable* if there is a single unsolvable quadratic inequality that can be obtained as a linear combination of the given inequalities. By the Farkas Lemma, the analogous condition for the solvability of a system of linear inequalities is necessary and sufficient. In the quadratic case, there are unsolvable but not strongly unsolvable systems. A nice example is given by the quadratic equations (14) and (16), and the linear equation $\sum_i x_i = k$. As we noted, this system is unsolvable for $k > \alpha(G)$. However, it can be shown that it is strongly unsolvable only for $k > \theta(G)$. So if we take G to be the pentagon and $k = 2.1$, we get an unsolvable, but not strongly unsolvable system.

Using semidefinite optimization, we get a solution for a very special but important case:

Theorem 7.3 *It is decidable in polynomial time whether a system of quadratic inequalities is strongly unsolvable.*

7.3 Inference rules

An *inference rule* for algebraic inequalities is a procedure that, given a system $\alpha_1 \geq 0, \dots, \alpha_m \geq 0$ of algebraic inequalities in n variables, determines a new algebraic inequality $\alpha \geq 0$, which is a *logical consequence* of the given system in the sense that every vector $x \in \mathbb{R}^n$ satisfying $\alpha_1(x) \geq 0, \dots, \alpha_m(x) \geq 0$ also satisfies $\alpha(x) \geq 0$. Perhaps the simplest inference rule is the following.

Linear combination rule:

$$\alpha_1 \geq 0, \dots, \alpha_m \geq 0 \implies c_0 + c_1\alpha_1 + \dots c_m\alpha_m \geq 0 \quad (c_0, c_1, \dots, c_m \geq 0). \tag{38}$$

The Farkas Lemma asserts that among linear inequalities, this single rule generates *all* logical consequences. As we have mentioned, it is not sufficient once we have quadratic inequalities; however, in this case we can formulate other inference rules.

Multiplication rule:

$$\alpha_1 \geq 0,\ \alpha_2 \geq 0 \implies \alpha_1\alpha_2 \geq 0. \tag{39}$$

Assume that the linear inequalities $0 \leq x_i \leq 1$ as well as the quadratic equations $x_i^2 = x_i$ are present. Under this assumption, one can formulate the following

Restricted multiplication rule:

$$\alpha \geq 0 \implies x_i\alpha \geq 0,\ (1 - x_i)\alpha \geq 0. \tag{40}$$

The following rule will provide the connection with semidefinite optimization:

SQUARE RULE:

$$\alpha \geq 0 \implies \alpha + \beta_1^2 + \ldots + \beta_m^2 \geq 0 \tag{41}$$

(where the β_i are arbitrary polynomials). We can consider the RESTRICTED SQUARE RULE where all the β_i are linear.

Finally, let us formulate one other rule:

DIVISION RULE:

$$\alpha_1 \geq 0,\ (1+\alpha_1)\alpha_2 \geq 0 \implies \alpha_2 \geq 0. \tag{42}$$

A further restriction is obtained when we are not allowed to use the commutativity of the variables. We'll only consider this in connection with the restricted multiplication and linear rules.

Artin's Theorem (see below) implies that these rules are sufficient to derive all consequences of a system of algebraic inequalities. In the case of interest for us, namely linear consequences of linear programs with 0-1 variables, we don't need all these rules to generate all the logical consequences of our starting system. In fact, the following is true [71, 72, 12]:

Theorem 7.4 *Starting with any system of linear inequalities and the equations $x_i^2 = x_i$, repeated application of the Linear rule and the Restricted multiplication rule (even with the further non-commutativity restriction) generates all linear inequalities valid for the 0-1 solutions, in at most n iterations.*

7.4 Deriving facets of the stable set polytope

Deriving a facet in n iterations (as guaranteed by Theorem 7.4) gives little information about it. We have seen in section 7.1 that the most important facets of the stable set polytope can be derived in just one or two iterations. It turns out that (for the stable set polytope) one can obtain reasonably good bounds on the number of iterations needed to derive a facet, in terms of other useful parameters.

Let $\sum_i a_i x_i \leq b$ be an inequality defining a facet of STAB(G); we assume that it is scaled so that the a_i are relatively prime integers. We define its *defect* as $\sum_i a_i - 2b$. The defect of an odd hole constraint is 1; the defect of a clique constraint (5) is $|B| - 2$. In the case of a facet defined by an α-critical graph G, this value is the *Gallai class number* $\delta(G) = |V(G)| - 2\alpha(G)$ of the graph.

Lemma 7.5 [72] *Let $\sum_i a_i x_i \leq b$ be a facet of* STAB(G). *Then*

$$\max\left\{\sum_i a_i x_i :\ x \in \mathrm{FSTAB}(G)\right\} = \frac{1}{2}\sum_i a_i.$$

It follows that the defect is non-negative, and in fact it can be characterized as twice the *integrality gap* between optimizing over STAB and FSTAB:

Corollary 7.6 *The defect of a facet $\sum_i a_i x_i \le b$ satisfies*

$$\begin{aligned}\sum_i a_i - 2b &= 2\max\left\{\sum_i a_i x_i :\ x \in \mathrm{FSTAB}(G)\right\} \\ &- 2\max\left\{\sum_i a_i x_i :\ x \in \mathrm{STAB}(G)\right\}.\end{aligned}$$

Graphs that are α-critical with bounded Gallai class number have a finite classification [63]. There is a similar classification of facets of STAB(G) with bounded defect [61].

The following theorem can be proved by calculations similar to those given in section 7.1 above.

Theorem 7.7 [71, 72] *Let G any graph, and let F be a facet of* STAB(G)*, defined by the inequality $\sum_i a_i x_i \le b$, with defect δ.*

(a) *Starting with the non-negativity constraints* (3) *and the edge constraints* (4)*, the facet F can be derived, using the Linear and Restricted Multiplication rules, in at most δ steps.*

(b) *Starting with the non-negativity constraints* (3) *and the edge constraints* (4)*, the facet F can be derived, using the Linear, Restricted Multiplication, and Restricted Square rules, in at most b steps.*

If we also use the square rule, then the derivation may be much faster. For example, to derive a k-clique constraint using the Linear and Restricted multiplication rules takes $k-2$ steps; with the Restricted square rule, it takes only one. It seems that all the known "nice" (polynomially separable, see below) classes of facets of the stable set polytope, with the exception of the "Edmonds facets" in the case of the matching polytope, can be derived by one or two rounds of applications of the Linear, Restricted Multiplication, and Square Rules.

7.5 A bit of real algebraic geometry

Finally, let us put these considerations into a more general context. A fundamental theorem in real algebraic geometry is Artin's Theorem:

Theorem 7.8 *A polynomial $f \in \mathbb{R}[x_1, \ldots, x_n]$ is nonnegative for all $(x_1, \ldots, x_n) \in \mathbb{R}^n$ if and only if it is a sum of squares of rational functions.*

One might expect that the term "rational functions" can be replaced by "polynomials", but this cannot be guaranteed in general. In special cases of combinatorial interest, however, we do get a simpler representation.

Let $G = (V, E)$ be a graph and let $I(G)$ denote the polynomial ideal generated by the polynomials $x_i^2 - x_i$ $(i \in V)$ and $x_i x_j$ $(ij \in E)$. Obviously, the roots of this ideal are the incidence vectors of stable sets. We write $f \ge 0 \pmod{I(G)}$ iff $f(x) \ge 0$ for every root of the ideal $I(G)$.

Proposition 7.9 *For any polynomial f, we have $f \geq 0 \pmod{I(G)}$ iff there exist polynomials $g_1, \ldots, g_N$ such that $f \equiv g_1^2 + \ldots + g_N^2 \pmod{I(G)}$.*

From theorem 5.13 it is easy to derive the following characterization of perfect graphs:

Theorem 7.10 *A graph G is perfect if and only if the following holds: For any linear polynomial f, we have $f \geq 0 \pmod{I(G)}$ iff there exist linear polynomials $g_1, \ldots, g_N$ such that $f \equiv g_1^2 + \ldots + g_N^2 \pmod{I(G)}$.*

7.6 Algorithmic aspects of inference rules

Let $\mathcal{L}$ be a possibly infinite system of linear inequalities in n variables, associated to a finite structure (e.g., a graph). We say that $\mathcal{L}$ is *polynomially separable*, if for every vector $x \in \mathbb{R}^n$, we can decide in polynomial time whether x satisfies every member of $\mathcal{L}$, and if it does not, we can find a violated member.

Let $\mathbf{R}$ be any inference rule, and let $\mathbf{R}\mathcal{L}$ denote the set of all linear inequalities produced by one application of $\mathbf{R}$ to members of $\mathcal{L}$. We say that the rule is *polynomial*, if $\mathbf{R}\mathcal{L}$ is polynomially separable whenever $\mathcal{L}$ is.

Using the ellipsoid method combined with semidefinite optimization, we get:

Lemma 7.11 *The Linear Rule* (38)*, the Restricted Multiplication Rule* (40) *and the Restricted Square Rule* (41) *are polynomial.*

It follows that if for some class of graphs, all facets of the stable set polytope can be derived by a bounded number of "rounds" of these three rules, then the stable set problem is polynomial for the class. In particular, we have the following consequences [42, 71, 72].

Corollary 7.12 *The Stable Set Problem can be solved for perfect, t-perfect and h-perfect graphs in polynomial time.*

Corollary 7.13 *Assume that for a class of graphs either the right hand side or the defect of each facet of the stable set polytope is bounded. Then the Stable Set Problem can be solved polynomially for this class.*

8 Extensions and problems

8.1 Small dimension representations and rank minimization

If we consider a semidefinite relaxation of a discrete optimization problem (say, a 0-1 linear program), then typically the original solutions correspond to semidefinite matrices of rank 1. In linear programming, there are special but useful conditions that guarantee that the solutions of the relaxed linear problem are also solutions of the original integer problem (for example, perfectness, or total unimodularity).

Problem 8.1 *Find combinatorial conditions that guarantee that the semidefinite relaxation has a solution of rank 1.*

This question can be interesting for special combinatorial semidefinite relaxations. For example,

Problem 8.2 *Which graphs are "max-cut-perfect?"*

Theorem 7.10 suggests an algebraic question:

Problem 8.3 *Which polynomial ideals I are "perfect" in the sense that for any linear polynomial f, we have $f \geq 0 \pmod I$ iff there exist linear polynomials $g_1, \ldots, g_N$ such that $f \equiv g_1^2 + \ldots + g_N^2 \pmod I$? Of course, there is a lot of room to modify the question by replacing "linear" with "bounded degree", etc.*

Coming back to semidefinite programs, if we find a solution that has, instead of rank 1, some other small rank, (i.e., a vector solution in low dimension), then this may decrease the error of the rounding methods, used to extract approximate solutions to the original problems. Thus the version of problem 8.1 with "low rank" instead of "rank 1" also seems very interesting. One result in this direction is the following (discovered in many versions [14, 36, 80, 59]; see also [27], section 31.5, and [15]):

Theorem 8.4 *The semidefinite system*

$$\begin{aligned} &X \succeq 0 \\ &D_1 \cdot X = d_1 \\ &\vdots \\ &D_k \cdot X = d_k, \end{aligned}$$

has a solution of rank at most $\lceil \sqrt{2k} \rceil$.

Also from a geometric point of view, it is natural to consider unit distance (orthogonal, etc.) representations in a fixed small dimension. Without control over the rank of the solutions of semidefinite programs, this additional condition makes the use of semidefinite optimization methods very limited. On the other hand, several of these geometric representations of graphs are connected to interesting graph-theoretic properties, and some of them are related to semidefinite optimization. This connection is largely unexplored.

Let us mention a few examples where we do have some information about low rank solutions. A vector labeling $V \to \mathbb{R}^d$ is *generic* if any d labels are linearly independent. Let $\kappa(G)$ denote the node-connectivity of G. The following was proved in [69] (see also [70]):

Theorem 8.5 *The minimum dimension in which a graph G has a generic orthogonal representation is $n - \kappa(G)$.*

In other words, the smallest d for which the semidefinite constraints

$$\begin{aligned} Y &\succeq 0 \\ Y_{ij} &= 0 \qquad \forall\, ij \notin E,\ i \neq j \end{aligned}$$

have a solution of rank d such that every $d \times d$ subdeterminant is non-zero, is exactly $n - \kappa(G)$.

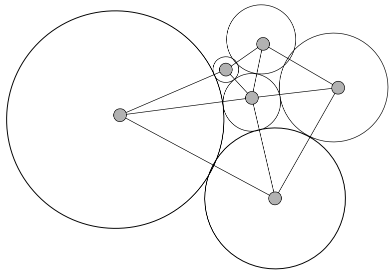

Figure 10: Representing a planar graph by touching circles

A classical result of Koebe [58] (see also [9, 91, 86], asserts that every planar graph can be represented in the plane by touching circular disks (Figure 10. One of the many extensions of this theorem characterizes triangulations of the plane that have a representation by orthogonal circles: more exactly, circles representing adjacent nodes must intersect at 90°, other pairs, at $> 90°$ (i.e., their centers must be farther apart) [9, 91, 56] (Figure 11.

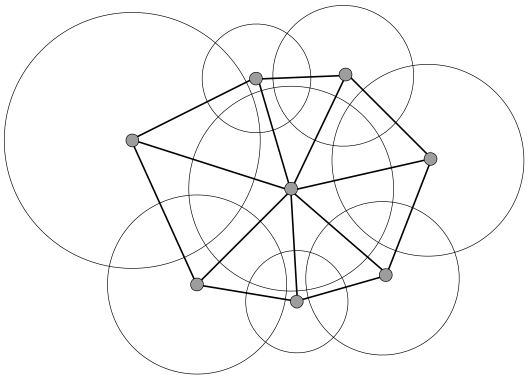

Figure 11: Representing a planar graph by orthogonal circles

Such a representation, if it exists, can be projected to a representation by orthogonal circles on the unit sphere; with a little care, one can do the projection so that each disk bounded by one of the circles is mapped onto a "cap" which covers less than half of the sphere. Then each cap has a unique *pole*: the point in space from which the part of the sphere you see is exactly the given cap. The key observation is that *two circles are orthogonal if and only if the corresponding poles have inner product* 1 (Figure 12). This translates a representation with orthogonal circles into a representation by vectors of length larger than 1, where adjacent nodes are represented by vectors with inner product 1, non-adjacent nodes by vectors with inner product less than 1.

This in turn can be translated into semidefinite matrices. We only state the final result of these transformations. Consider the following two sets of semidefinite constraints:

$$\begin{aligned} Y &\succeq 0 \\ Y_{ij} &= 1 \qquad \forall\, ij \in E, \\ Y_{ij} &< 1 \qquad \forall\, ij \notin E,\ i \neq j, \end{aligned} \tag{43}$$

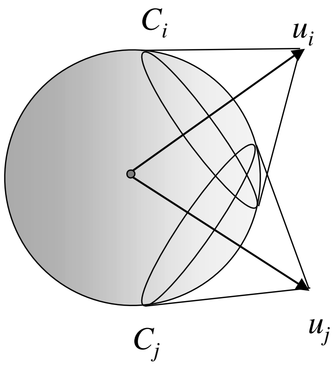

Figure 12: Poles of circles

$$Y_{ii} > 1$$

and the weaker set of constraints

$$\begin{aligned} Y &\succeq 0 \\ Y_{ij} &= 1 \qquad \forall\, ij \in E, \\ Y_{ij} &< 1 \qquad \forall\, ij \notin E,\ i \neq j, \end{aligned} \tag{44}$$

(45)

To formulate the theorem, we need two simple definitions. A cycle C in a graph G is called *separating*, if $G \setminus V(C)$ has at least two connected components, where any chord of C is counted as a connected component here. The cycle C is called *strongly separating*, if $G \setminus V(C)$ has at least two connected components, each of which has at least 2 nodes. If G is a 3-connected planar map, then its non-separating cycles are exactly the boundaries of the faces.

Theorem 8.6 *Let G be a 3-connected graph*

(a) *If (44) has a solution of rank 3, then G is planar.*

(b) *Assume that G is a maximal planar graph. Then (43) has a solution of rank 3 if and only if G has no separating 3- and 4-cycles.*

(c) *Assume that G is a maximal planar graph. Then (44) has a solution with rank 3 if and only if G has no strongly separating 3- and 4-cycles.*

Colin de Verdière [24] introduced an interesting spectral invariant of graphs that is related to topological properties. Kotlov, Lovász and Vempala [56] showed that this invariant can be defined in terms of the minimum rank of a “non-degenerate” solution of (44) (see [3] for the definition and theory of non-degeneracy in semidefinite programs).

Tutte [92] constructed a straight-line embedding in the plane of a 3-connected planar graph by fixing the vertices of a face to the vertices of a convex polygon, replacing the edges by "rubber bands", and letting the other nodes find their equilibrium (Figure 13). A similar construction was used in [60] to characterize k-connectivity of a graph, and to

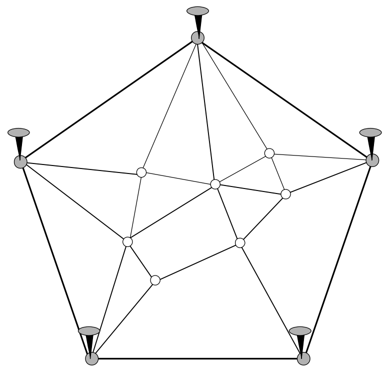

Figure 13: Tutte's "rubber band" representation of planar graphs

design an efficient randomized k-connectivity test. There is an obvious similarity with our description of the Goemans-Williamson algorithm in the introduction, and we could obtain the equilibrium situation through a semidefinite program. But in Tutte's case the sum of squares of edge lengths is to be minimized, rather than maximized; since this function is concave, this makes a substantially better behaved optimization problem, which can be solved efficiently in every fixed dimension. What is important for us, however, is that this is an example of a semidefinite program whose solution has fixed small rank.

Rubber band problems form a special class of semidefinite optimization problems which can be solved by direct means. Further such problems are described in [95]. It would be interesting to understand the structure of such special classes.

A final remark: many problems in graph theory, matroid theory, electrical engineering, statics etc. can be formulated as *maximizing* the rank of a matrix subject to linear constraints (see [84, 66]). Such problems can be solved by an obvious polynomial time randomized algorithm, by substituting random numbers for the variables. Unlike in the case of the randomized algorithms described above for the Max Cut and other problems, it is not known whether these rank maximization problems can be solved in deterministic polynomial time.

8.2 Approximation algorithms

The most important open question is: can the randomized "rounding" method of Goemans–Williamson and Karger–Motwani–Sudan be generalized to semidefinite relaxations of more general problems? Can other, different rounding techniques be found?

There are many candidate problems, the most interesting is the "class of the factor 2". We have seen that the Maximum Cut problem has a trivial factor 2 approximation algorithm. There are several other such optimization problems; here are three very fundamental examples:

The Node Cover problem: given a graph G, find a minimum set of nodes covering all edges.

The Acyclic Subgraph problem: given a directed graph, find the maximum number of edges that form no directed cycle.

The Overdetermined Binary Equations problem: given a system of linear equations over GF(2), find an assignment of the variables that satisfies as many of them as possible.

We leave it to the reader to find the easy algorithms that give suboptimal solutions off by a factor of 2 or less. In all cases it is known that we cannot bring this error factor arbitrarily close to 1.

Problem 8.7 Can we do better than the trivial factor of 2?

In the case of the Maximum Cut problem, we saw that the answer is positive. Surprisingly, for the Overdetermined Binary Equations problem (which is in fact a generalization of the Maximum Cut problem) Håstad [45] showed that the answer is negative: the factor of 2 is optimal. For the Node Cover and Acyclic Subgraph problems the question is open. The most promising technique to attack these questions is semidefinite optimization, even though the attempts by many have not been successful so far.

There are many open questions about approximating the stability number (or equivalently, the largest clique), and the chromatic number (whether or not semidefinite optimization can be used in answering these is not clear):

Problem 8.8 Can the ratio ϑ/α be estimated by $n^{1-\varepsilon}$ for special classes of graphs? Are there interesting classes of graphs for which the ϑ can be bounded by some function (or small function) of α?

Problem 8.9 Can $\alpha(G)$ be approximated better than the error factor $n/(\log n)^2$ (this is achieved in [21]).

Problem 8.10 Is there a polynomial time algorithm that outputs an upper bound $\phi(G)$ for $\alpha(G)$ such that there is a function $f : \mathbb{Z}_+ \to \mathbb{Z}_+$ with $\phi(G) < f(\alpha(G))$ (f is independent of the size of the graph)?

Problem 8.11 Is is true that for every $\varepsilon > 0$ there exists an algorithm that computes $\alpha(G)$ in time $O((1+\varepsilon)^n)$?

Problem 8.12 Suppose that G is a graph with chromatic number 3. Can G be k-colored in polynomial time, where (a) $k = n^{o(1)}$; (b) $k = \log n$; (c) $k = O(1)$?

8.3 Inference rules

We discussed strong insolvability of systems of quadratic equations. Barvinok [13] gives a polynomial time algorithm to decide whether a system of a bounded number of quadratic equations is solvable (over the real field). This suggests a hierarchy of extensions of strong insolvability: produce a fixed number k of quadratic equations by linear combination which are collectively unsolvable.

Problem 8.13 Can one decide in polynomial time the k-th version of strong insolvability? Is this a real hierarchy? Are there any natural problems in higher classes?

Problem 8.14 Are the multiplication rule (39) and the division rule (42) polynomial? Are they polynomial if we restrict ourselves to quadratic inequalities? If not, does the division rule have a natural and useful restriction that is polynomial?

Problem 8.15 Are there other combinatorial optimization problems for which interesting classes of facets can be derived using the division rule?

Problem 8.16 Are there other inference rules that are worth considering? Can any interesting discrete programming problem be attacked using polynomials of higher degree?

Problem 8.17 How to implement the restricted multiplication rule (40) efficiently? Is there a way to use interior point methods, in a way parallel to Alizadeh's application of interior point methods to semidefinite programming?

Problem 8.18 If a graph G contains no subdivision of K_4, then it is series-parallel, and hence t-perfect [22]. This means that every facet of $\mathrm{STAB}(G)$ has defect at most 1. Is there an analogous simple graph-theoretic condition that guarantees that every facet has defect at most 2, 3, etc.?

Acknowledgement. My thanks are due to András Frank and to Bruce Reed for organizing two series of talks on this subject. I am particularly indebted to Miklós Újváry for pointing out several errors in an earlier version of these notes, and to Wayne Barrett and the anonymous referees of this version for suggesting many improvements.

References

[1] F. Alizadeh: Combinatorial optimization with semi-definite matrices, in: *Integer Programming and Combinatorial Optimization* (Proceedings of IPCO '92), (eds. E. Balas, G. Cornuéjols and R. Kannan), Carnegie Mellon University Printing (1992), 385–405.

[2] F. Alizadeh, Interior point methods in semidefinite programming with applications to combinatorial optimization, *SIAM J. Optim.* **5** (1995), 13–51.

[3] F. Alizadeh, J.-P. Haeberly, and M. Overton: Complementarity and nondegeneracy in semidefinite programming, in: *Semidefinite Programming*, *Math. Programming* Ser. B, **77** (1997), 111–128.

[4] N. Alon, R. A. Duke, H. Lefmann, V. Rödl and R. Yuster: The algorithmic aspects of the Regularity Lemma, *Proc. 33rd Annual Symp. on Found. of Computer Science*, IEEE Computer Society Press (1992), 473–481.

[5] N. Alon and J.H. Spencer: *The Probabilistic Method*, Wiley, New York, 1992.

[6] N. Alon, The Shannon capacity of a union, *Combinatorica* **18** (1998), 301–310.

[7] N. Alon: Explicit Ramsey graphs and orthonormal labelings, *The Electronic Journal of Combinatorics* **1** (1994), 8pp.

[8] N. Alon and N. Kahale: Approximating the independence number via the ϑ-function, *Math. Programming* **80** (1998), Ser. A, 253–264.

[9] E. Andre'ev, On convex polyhedra in Lobachevsky spaces, *Mat. Sbornik*, Nov. Ser. **81** (1970), 445–478.

[10] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy: Proof verification and hardness of approximation problems *Proc. 33rd FOCS* (1992), 14–23.

[11] R. Bacher and Y. Colin de Verdière, Multiplicités de valeurs propres et transformations étoile-triangle des graphes, *Bull. Soc. Math. France* **123** (1995), 101-117.

[12] E. Balas, S. Ceria and G. Cornuéjols, A lift-and-project cutting plane algorithm for mixed $0-1$ programs, *Mathematical Programming* **58** (1993), 295–324.

[13] A.I. Barvinok: Feasibility testing for systems of real quadratic equations, *Discrete and Comp. Geometry* **10** (1993), 1–13.

[14] A.I. Barvinok: Problems of distance geometry and convex properties of quadratic maps, *Discrete and Comp. Geometry* **13** (1995), 189–202.

[15] A.I. Barvinok: A remark on the rank of positive semidefinite matrices subject to affine constraints, *Discrete and Comp. Geometry* **25** (2001), 23–31.

[16] J. Beck: Roth's estimate on the discrepancy of integer sequences is nearly sharp, *Combinatorica* **1** (1981) 327–335.

[17] J. Beck and W. Chen: *Irregularities of Distribution,* Cambridge Univ. Press (1987).

[18] J. Beck and V.T. Sós: Discrepancy Theory, Chapter 26 in: *Handbook of Combinatorics* (ed. R.L. Graham, M. Grötschel and L. Lovász), North-Holland, Amsterdam (1995).

[19] M. Bellare, O. Goldreich, M. Sudan: Free bits, PCPs and non-approximability — towards tight results, *Proc. 36th FOCS* (1996), 422–431.

[20] A. Blum and D. Karger: An $O(n^{3/14})$-coloring for 3-colorable graphs, *Inform. Process. Lett.* **61** (1997), 49–53.

[21] R. Boppana and M. Haldórsson: Approximating maximum independent sets by excluding subgraps, *BIT* **32** (1992), 180–196.

[22] M. Boulala and J.-P. Uhry: Polytope des indépendants d'un graphe série-parallèle, *Discrete Math.* **27** (1979), 225–243.

[23] V. Chvátal: On certain polytopes associated with graphs, *J. of Combinatorial Theory (B)* **18** (1975), 138–154.

[24] Y. Colin de Verdière, Sur la multiplicité de la première valeur propre non nulle du laplacien, *Comment. Math. Helv.* **61** (1986), 254–270.

[25] Y. Colin de Verdière, Sur un novel invariant des graphes at un critère de planarité, *J. Combin. Theory B* **50** (1990) 11–21.

[26] Y. Colin de Verdière, On a new graph invariant and a criterion for planarity, in: *Graph Structure Theory* (Robertson and P. D. Seymour, eds.), Contemporary Mathematics, Amer. Math. Soc., Providence, RI (1993), 137–147.

[27] M. Deza and M. Laurent: *Geometry of Cuts and Metrics*, Springer Verlag, 1997.

[28] C. Delorme and S. Poljak: Combinatorial properties and the complexity of max-cut approximations, *Europ. J. Combin.* **14** (1993), 313–333.

[29] C. Delorme and S. Poljak: Laplacian eigenvalues and the maximum cut problem, *Math. Programming* **62** (1993)

[30] P. Erdős: Gráfok páros körüljárású részgráfjairól (On bipartite subgraphs of graphs, in Hungarian), *Mat. Lapok* **18** (1967), 283–288.

[31] P. Erdős, F. Harary and W.T. Tutte, On the dimension of a graph *Mathematika* **12** (1965), 118–122.

[32] U. Feige: Randomized graph products, chromatic numbers, and the Lovász ϑ-function, *Combinatorica* **17** (1997), 79–90.

[33] U. Feige: Approximating the Bandwidth via Volume Respecting Embeddings, Tech. Report CS98-03, Weizmann Institute (1998).

[34] U. Feige and M. Goemans, Approximating the value of two-prover proof systems, with applications to MAX-2SAT and MAX-DICUT, in: Proc. 3rd Israel Symp. on Theory and Comp. Sys., Tel Aviv, Isr. (1995), 182–189.

[35] U. Feige and L. Lovász: Two-prover one-round proof systems: Their power and their problems. *Proc. 24th ACM Symp. on Theory of Computing* (1992), 733-744.

[36] S. Friedland and R. Loewy, Subspaces of symmetric matrices containing matrices with multiple first eigenvalue, *Pacific J. Math.* **62** (1976), 389–399.

[37] M. X. Goemans and D. P. Williamson: .878-Approximation algorithms for MAX CUT and MAX 2SAT, *Proc. 26th ACM Symp. on Theory of Computing* (1994), 422-431.

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

[39] M. C. Golumbic: *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York (1980).

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

[41] M. Grötschel, L. Lovász and A. Schrijver: Polynomial algorithms for perfect graphs, *Annals of Discrete Math.* **21** (1984), 325-256.

[42] M. Grötschel, L. Lovász and A. Schrijver: Relaxations of vertex packing, *J. Combin. Theory B* **40** (1986), 330-343.

[43] M. Grötschel, L. Lovász and A. Schrijver: *Geometric Algorithms and Combinatorial Optimization*, Springer, Heidelberg, 1988.

[44] W. Haemers: On some problems of Lovász concerning the Shannon capacity of a graph, *IEEE Trans. Inform. Theory* **25** (1979), 231–232.

[45] J. Håstad: Some optimal in-approximability results, *Proc. 29th ACM Symp. on Theory of Comp.*, 1997, 1–10.

[46] J. Håstad: Clique is hard to approximate within a factor of $n^{1-\varepsilon}$, *Acta Math.* **182** (1999), 105–142.

[47] H. van der Holst, A short proof of the planarity characterization of Colin de Verdière, Preprint, CWI Amsterdam, 1994.

[48] H. van der Holst, L. Lovász and A. Schrijver: On the invariance of Colin de Verdière's graph parameter under clique sums, *Linear Algebra and its Applications*, **226–228** (1995), 509–518.

[49] F. Juhász: The asymptotic behaviour of Lovász' ϑ function for random graphs, *Combinatorica* **2** (1982) 153–155.

[50] N. Kahale: A semidefinite bound for mixing rates of Markov chains, DIMACS Tech. Report No. 95-41.

[51] D. Karger, R. Motwani, M. Sudan: Approximate graph coloring by semidefinite programming, *Proc. 35th FOCS* (1994), 2–13; full version: *J. ACM* **45** (1998), 246–265.

[52] H. Karloff: How good is the Goemans-Williamson MAX CUT algorithm? *SIAM J. Comput.* **29** (1999), 336–350.

[53] H. Karloff and U. Zwick: A 7/8-approximation algorithm for MAX 3SAT? in: *Proc. of the 38th Ann. IEEE Symp. in Found. of Comp. Sci.* (1997), 406–415.

[54] B. S. Kashin and S. V. Konyagin: On systems of vectors in Hilbert spaces, *Trudy Mat. Inst. V.A.Steklova* **157** (1981), 64–67; English translation: *Proc. of the Steklov Inst. of Math.* (AMS 1983), 67–70.

[55] V. S. Konyagin, Systems of vectors in Euclidean space and an extremal problem for polynomials, *Mat. Zametky* **29** (1981), 63–74. English translation: *Math. Notes of the Academy USSR* **29** (1981), 33–39.

[56] A. Kotlov, L. Lovász, S. Vempala, The Colin de Verdière number and sphere representations of a graph, *Combinatorica* **17** (1997) 483–521.

[57] D. E. Knuth: The sandwich theorem, *The Electronic Journal of Combinatorics* **1** (1994) 48 pp.

[58] P. Koebe: Kontaktprobleme der konformen Abbildung, *Berichte uber die Verhandlungen d. Sächs. Akad. d. Wiss.*, Math.–Phys. Klasse, **88** (1936) 141–164.

[59] M. Laurent and S. Poljak: On the facial structure of the set of correlation matrices, *SIAM J. on Matrix Analysis and Applications* **17** (1996), 530–547.

[60] N. Linial, L. Lovász, A. Wigderson: Rubber bands, convex embeddings, and graph connectivity, *Combinatorica* **8** (1988), 91–102.

[61] L. Lipták, L. Lovász: Facets with fixed defect of the stable set polytope, *Math. Programming*, Series A **88** (2000), 33–44.

[62] L. Lovász: Normal hypergraphs and the perfect graph conjecture, *Discrete Math.* **2** (1972), 253-267.

[63] L. Lovász: Some finite basis theorems in graph theory, in: *Combinatorics*, Coll. Math. Soc. J. Bolyai **18** (1978), 717-729.

[64] L. Lovász: On the Shannon capacity of graphs, *IEEE Trans. Inform. Theory* **25** (1979), 1–7.

[65] L. Lovász: Perfect graphs, in: *More Selected Topics in Graph Theory* (ed. L. W. Beineke, R. L. Wilson), Academic Press (1983), 55-67.

[66] L. Lovász: Singular spaces of matrices and their applications in combinatorics, *Bol. Soc. Braz. Mat.* **20** (1989), 87–99.

[67] L. Lovász: Stable sets and polynomials, *Discrete Math.* **124** (1994), 137–153.

[68] L. Lovász: Integer sequences and semidefinite programming *Publ. Math. Debrecen* **56** (2000) 475–479.

[69] L. Lovász, M. Saks and A. Schrijver: Orthogonal representations and connectivity of graphs, *Linear Alg. Appl.* **114/115** (1989), 439–454.

[70] L. Lovász, M. Saks and A. Schrijver: A correction: orthogonal representations and connectivity of graphs (with M. Saks and A. Schrijver) *Linear Algebra Appl.* **313** (2000) 101–105.

[71] L. Lovász and A. Schrijver: Cones of matrices and set-functions, and 0-1 optimization, *SIAM J. on Optimization* **1** (1990), 166-190.

[72] L. Lovász and A. Schrijver: Matrix cones, projection representations, and stable set polyhedra, in: *Polyhedral Combinatorics*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science I, Amer. Math. Soc., Providence (1990), 1–17.

[73] L. Lovász and K. Vesztergombi: Geometric representations of graphs, in: *Paul Erdős and his Mathematics*

[74] J. Matoušek and J. Spencer, Discrepancy in arithmetic progressions, *J. Amer. Math. Soc.* **9** (1996) 195–204.

[75] B. Mohar and S. Poljak: Eigenvalues and the max-cut problem, *Czechoslovak Mathematical Journal* **40** (1990), 343–352.

[76] Yu. E. Nesterov and A. Nemirovsky: *Interior-point polynomial methods in convex programming,* Studies in Appl. Math. **13**, SIAM, Philadelphia, 1994.

[77] M. L. Overton: On minimizing the maximum eigenvalue of a symmetric matrix, *SIAM J. on Matrix Analysis and Appl.* **9** (1988), 256–268.

[78] M. L. Overton and R. Womersley: On the sum of the largest eigenvalues of a symmetric matrix, *SIAM J. on Matrix Analysis and Appl.* **13** (1992), 41–45.

[79] M. Padberg: *Linear optimization and extensions.* Second, revised and expanded edition, Algorithms and Combinatorics **12**, Springer-Verlag, Berlin, 1999.

[80] G. Pataki: On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues, *Math. of Oper. Res.* **23** (1998), 339–358.

[81] S. Poljak and F. Rendl: Nonpolyhedral relaxations of graph-bisection problems, DIMACS Tech. Report 92-55 (1992).

[82] L. Porkoláb and L. Khachiyan: On the complexity of semidefinite programs, *J. Global Optim.* **10** (1997), 351–365.

[83] M. Ramana: An exact duality theory for semidefinite programming and its complexity implications, in: *Semidefinite programming. Math. Programming* Ser. B, **77** (1997), 129–162.

[84] A. Recski: *Matroid Theory and its Applications in Electric Network Theory and Statics*, Akadémiai Kiadó–Springer-Verlag (1989).

[85] K.F. Roth: Remark concerning integer sequences, *Acta Arith.* **35**, 257–260.

[86] O. Schramm: How to cage an egg, *Invent. Math.* **107** (1992), 543–560.

[87] H.D. Sherali and W.P. Adams (1990): A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems, *SIAM J. on Discrete Math.* bf 3, 411–430.

[88] G. Strang: *Linear algebra and its applications*, Second edition, Academic Press, New York–London, 1980.

[89] M. Szegedy: A note on the θ number of Lovász and the generalized Delsarte bound, *Proc. 35th FOCS* (1994), 36–39.

[90] É. Tardos: The gap between monotone and non-monotone circuit complexity is exponential, *Combinatorica* **8** (1988), 141–142.

[91] W. Thurston: *The Geometry and Topology of Three-manifolds*, Princeton Lecture Notes, Chapter 13, Princeton, 1985.

[92] W.T. Tutte: How to draw a graph, *Proc. London Math. Soc.* **13** (1963), 743-768.

[93] L. Vandeberghe and S. Boyd: Semidefinite programming, in: *Math. Programming: State of the Art* (ed. J. R. Birge and K. G. Murty), Univ. of Michigan, 1994.

[94] L. Vandeberghe and S. Boyd: Semidefinite programming. SIAM Rev. 38 (1996), no. 1, 49–95.

[95] R.J. Vanderbei and B. Yang: The simplest semidefinite programs are trivial, *Math. of Oper. Res.* **20** (1995), no. 3, 590–596.

[96] H. Wolkowitz: Some applications of optimization in matrix theory, *Linear Algebra and its Applications* **40** (1981), 101–118.

[97] H. Wolkowicz: Explicit solutions for interval semidefinite linear programs, *Linear Algebra Appl.* **236** (1996), 95–104.

[98] H. Wolkowicz, R. Saigal and L. Vandenberghe: *Handbook of semidefinite programming. Theory, algorithms, and applications.* Int. Ser. Oper. Res. & Man. Sci., **27** (2000) Kluwer Academic Publishers, Boston, MA.

[99] U. Zwick: Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to MAX CUT and other problems, *Proc. 31th STOC* (1999), 679–687.