Lecture 9

Rounding MAXQP

9.1 General quadratic programs

Consider the following problem

\[ \text{MAXQP}(A) = \sup_{x, y \in \{\pm 1\}} \sum_{i,j} A_{i,j} x_i y_j, \]

where \( A \in \mathbb{R}^{n \times m} \).

**Exercise 1.** Show that this problem is NP-hard by performing a reduction from MAXCUT.

We will see how a good approximation can be obtained in polynomial time. For this we propose the following relaxation:

\[ \text{MAXQP}(G) \leq \text{SDP}(G) = \sup_{u, v \in \mathbb{R}^{m+n}} \sum_{i,j} A_{i,j} u_i \cdot v_j. \quad (9.1) \]

Since the \( u_i \) and \( v_j \) are different sets of vectors it may not be immediately obvious that this program is an SDP, but it is — we will show this soon. The following theorem guarantees that the above relaxation is never too bad — SDP(G) is never more than a constant factor (independent of \( A \) and its dimension) larger than MAXQP(G). Moreover we’ll give a constructive proof of this fact, showing how any vector solution to the SDP can be “rounded” to a ±1 solution to the original integer program.

**Theorem 9.1.** Given unit vectors \( u_i \) and \( v_j \) achieving the optimum in SDP(G), there exists a polynomial-time algorithm that produces \( x_i \) and \( y_j \) in \( \{\pm 1\} \) such that

\[ \sum_{i,j} A_{i,j} x_i y_j \geq C \cdot \text{MAXQP}(G), \]

where \( C \) is a universal constant.
In fact, we’ll show a bit more, we’ll prove that \( \sum_{i,j} A_{i,j} x_i y_j \geq C \cdot \text{SDP}(G) \), which of course is at least \( C \cdot \text{MAXQP}(G) \).

Different proof techniques for the theorem yield different values of \( C \). In your homework you will develop an algorithm to achieve \( C \approx 0.56 \). The best value for \( C \) is called Grothendieck’s constant \( K_G \) and can be defined as

\[
K_G = \inf \left\{ C : \forall m, n, \forall A \in \mathbb{R}^{n \times m}, \sup_{i,j} A_{i,j} u_i \cdot v_j \leq K_G \sup_{i,j} A_{i,j} x_i y_j \right\}
\]

Before proceeding let’s check that SDP(\( A \)) is indeed an SDP by writing it in the primal canonical form

\[
\sup \ B \cdot Z \\
\text{s.t.} \quad A_i \cdot Z \leq c_i \\
Z \succ 0.
\]

Let \( U \) be a matrix whose columns are the \( u_i \), and \( V \) whose columns are the \( v_j \); define

\[
Z = (UV)^T(UV) = \begin{pmatrix} [u_i \cdot u_j] & [u_i \cdot v_j] \\ [v_i \cdot u_j] & [v_i \cdot v_j] \end{pmatrix} \in \mathbb{R}^{(m+n) \times (m+n)}.
\]

Clearly \( Z \) is a PSD matrix (it is a Gram matrix), and its diagonal elements are the squared norms \( \|u_i\|^2 \) and \( \|v_j\|^2 \), which should be at most one. Thus we let \( c_i = 1 \) and \( A_i = E_{ii} \) for \( i = 1, \ldots, n+m \), where \( E_{ii} \) is a \( (n+m) \times (n+m) \) matrix whose \( i \)th diagonal entry is one and the others are zero. Finally for the objective value, we define

\[
B = \frac{1}{2} \begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}.
\]

Then the corresponding SDP is equivalent to the definition of SDP(\( A \)) given earlier.

### 9.2 Analysis of the SDP relaxation

In this section we analyze the performance of the relaxation (9.1). The main result is that we can bound the ratio \( \text{SDP}(A)/\text{MAXQP}(A) \) by a constant factor, regardless of the choice of \( A \).

**Theorem 9.2.** There exists a universal constant \( K = K_G \) such that \( \forall A, \text{SDP}(A) \leq K \cdot \text{MAXQP}(A) \).

**Remark 9.3.** (1) If one does not care about the optimal constant the result can be made “algorithmic” in the following sense: the proof we’ll give in this lecture shows that given any unit vectors \( u_i, v_j \) achieving SDP(\( A \)), there is a randomized (even deterministic) polynomial-time algorithm that outputs \( x_i, y_j \in \{\pm 1\} \) such that

\[
\sum_{i,j} A_{i,j} x_i y_j \geq \alpha \sum_{i,j} A_{i,j} u_i \cdot v_j
\]

for some \( \alpha \geq 0.01 \).
(2) One can do better in the value of $\alpha$. The best possible constant is known as Grothendieck’s constant, which is strictly greater than 0.56. We do not know its precise value — only the first few digits have been pinned down.

Let’s prove Theorem 9.2. Let $u_i, v_j \in \mathbb{R}^d$ achieving SDP$(A)$ be given (note that technically we should assume they achieve SDP$(A) - \varepsilon$, as we don’t know if the supremum is attained; for clarity we’ll put this issue aside). We can always assume $d \leq m + n$ since this is the number of vectors. Observe that

$$\sum_{i,j} A_{ij} u_i \cdot v_j = \frac{1}{d} \sum_{k=1}^d \left( \sum_{i,j} A_{ij} \left( \sqrt{d}(u_i)_k \right) \left( \sqrt{d}(v_j)_k \right) \right).$$

This implies we can find $k \in \{1, \ldots, d\}$ such that

$$\sum_{i,j} A_{ij} \left( \sqrt{d}(u_i)_k \right) \left( \sqrt{d}(v_j)_k \right) \geq \sum_{i,j} A_{ij} u_i \cdot v_j = \text{SDP}(A) \geq \text{OPT}(A).$$

How large are the $|(u_i)_k|$? From $\|u_i\| \leq 1$ we know $|(u_i)_k| \leq 1$, which implies $\sqrt{d}(u_i)_k \leq \sqrt{d}$. This naive bound is not good enough: we are looking for an assignment of values in the range $[-1, 1]$, so we’d have to divide all coordinates by $\sqrt{d}$, but then we’d lose a factor $d$ in the objective value.

Now, if everything was “well-behaved”, i.e. the vectors are random, we would expect $|(u_i)_k| \approx 1/\sqrt{d}$ because $\|u_i\| \leq 1$ — in this case barely any renormalization is needed. This heuristic suggests the approach for the proof, which is to perform a “random” rotation (we will in fact apply a deterministic transformation) that guarantees the coordinates are well-balanced, so that most of them are not too large, no larger than some constant. We’ll then “truncate” the coordinates that are too large, and argue that the loss in objective value suffered through this truncation is not too large.

Recall from Lecture 1 that we call a family of vectors $\vec{g}_1, \ldots, \vec{g}_t \in \mathbb{R}^d$ $k$-wise independent if for any $k$ distinct coordinates $j_1, \ldots, j_k \in \{1, \ldots, d\}$, the random variables $(X_{j_1}, \ldots, X_{j_k})$ defined by selecting a random $i \in \{1, \ldots, t\}$ and returning $(\vec{g}_i)_{j_1}, \ldots, (\vec{g}_i)_{j_k}$ are independent. Say furthermore that the vectors are uniformly $k$-wise independent if the random variables we just defined are uniformly distributed.

In your homework you proved the following claim using a construction based on Vandermonde matrices:

**Claim 9.4.** In dimension $d$, there exists $t = O(d^2)$ vectors $\vec{g}_1, \ldots, \vec{g}_t \in \{\pm 1\}^d$, such that the $\vec{g}_i$ are uniformly 4-wise independent.

The claim can be interpreted as follows. Consider the matrix

$$G = \begin{pmatrix} -\vec{g}_1 & - \\ -\vec{g}_2 & - \\ \vdots \\ -\vec{g}_t & - \end{pmatrix}.$$
Then if we fix any 4 columns of $G$, corresponding to coordinates of the $\vec{g}_i$, each possible pattern in $\{\pm 1\}^4$ occurs exactly $t/16$ times. This property naturally implies $r$-wise independence for $r < 4$. That is, if we look at a single column, the number of 1’s should be the same as the number of −1’s, and similar conditions hold for two and three columns. Note that if we allow exponential large $t$, we can just choose all the possible $d$-dimensional ±1 vectors to satisfy this property. The main point here is that we can make this $t$ only polynomial in $d$.

### 9.3 Rounding MAXQP

Let’s get to the proof of Theorem 9.2: we need to show that there exists a universal constant $K > 0$ such that for all $A \in \mathbb{R}^{m \times n}$,

$$\text{MAXQP}(A) = \max_{x_i, y_j \in \{\pm 1\}} \sum A_{ij} x_i y_j \leq \sup_{u_i, v_j \in \mathbb{R}^{n+m}, \|u_i\|, \|v_j\| \leq 1} \sum A_{ij} u_i \cdot v_j =: \text{SDP}(A) \leq \frac{1}{K} \text{MAXQP}(A).$$

Our technique for this will be to start with an optimal vector solution to SDP($A$), perform a (deterministic!) embedding of the vectors in a higher-dimensional space, and finally find a “good coordinate” to perform the rounding in the higher-dimensional space.

The embedding is defined as follows. For any $\vec{u} \in \mathbb{R}^d$, define $h(\vec{u}) \in \mathbb{R}^t$, such that $h(\vec{u})_i = \frac{\vec{g}_i \cdot \vec{u}}{\sqrt{t}} = (G\vec{u}/\sqrt{t})_i$ for $i = 1, \ldots, t$. For $M > 0$ define

$$h^M(\vec{u})_i = \begin{cases} h(\vec{u})_i & \text{if } |h(\vec{u})_i| \leq M/\sqrt{t}, \\ M/\sqrt{t} & \text{if } h(\vec{u})_i > M/\sqrt{t}, \\ -M/\sqrt{t} & \text{if } h(\vec{u})_i < -M/\sqrt{t}. \end{cases}$$

Then we have the following lemma:

**Lemma 9.5.** For any $\vec{u}, \vec{v} \in \mathbb{R}^d$, with $\|\vec{u}\| = \|\vec{v}\| = 1$,

1. $h(\vec{u}) \cdot h(\vec{v}) = \vec{u} \cdot \vec{v}$.
2. $\|h(\vec{u})\| = 1$.
3. $\|h^M(\vec{u})\| \leq 1$.
4. $\|h(\vec{u}) - h^M(\vec{u})\| \leq \frac{\sqrt{t}}{M}$.

We’ll prove the lemma later, first let’s use it to finish the proof of the theorem. The last property gives a precise trade-off between the size of $M$ and the quality of the approximation.
of the vector $h(\vec{u})$ by its truncation $h^M(\vec{u})$. Using the lemma, we can write

$$\text{SDP}(A) = \sum_{i,j} A_{ij} \vec{u}_i \cdot \vec{v}_j$$

$$= \sum_{i,j} A_{ij} h(\vec{u}_i) \cdot h(\vec{v}_j)$$

Using the lemma, we can write

$$\sum_{i,j} A_{ij} \vec{u}_i \cdot \vec{v}_j = \sum_{i,j} A_{ij} h(\vec{u}_i) \cdot h(\vec{v}_j)$$

Thus we can find a $k$ such that

$$\sum_{i,j} A_{ij} (h^M(\vec{u}_i))_k \cdot \frac{\sqrt{t}}{\sqrt{M}} (h^M(\vec{v}_j))_k \cdot \frac{\sqrt{t}}{\sqrt{M}} \geq \frac{1}{M^2} \text{SDP}(A) - \frac{2\sqrt{3}}{M} \text{SDP}(A).$$

To conclude, we set $M = 4$, $x_i = (h^M(\vec{u}_i))_k \cdot \frac{\sqrt{t}}{\sqrt{M}}$, $y_j = (h^M(\vec{v}_j))_k \cdot \frac{\sqrt{t}}{\sqrt{M}}$ and get

$$\sum_{i,j} A_{ij} x_{ik} y_{jk} \geq \text{SDP}(A) \cdot \frac{M - 2\sqrt{3}}{M^3} \approx 0.01 \cdot \text{SDP}(A).$$

Here $x_i, y_j \in [-1, 1]$, and we can always find values in $\{\pm 1\}$ that are at least as good (to see how, fix all the $x_i$ and observe that there is always an optimal setting of each individual $y_j$ that is either $+1$ or $-1$).

Thus the theorem is proved, conditioned on the lemma, whose proof we now turn to.

**Proof of Lemma 9.5.** The proof of the lemma relies on the 4-wise independence condition on the vectors $\vec{g}_i$. 


(1) We check that
\[
    h(\vec{u}) \cdot h(\vec{v}) = \sum_{k=1}^{t} \frac{\vec{g}_k \cdot \vec{u} \cdot \vec{g}_k \cdot \vec{v}}{\sqrt{t} \sqrt{t}}
\]
\[
    = \sum_{k=1}^{t} \sum_{i,j=1}^{d} \frac{(\vec{g}_k)_i \cdot \vec{u}_i (\vec{g}_k)_j \cdot \vec{v}_j}{\sqrt{t} \sqrt{t}}
\]
\[
    = \sum_{i,j=1}^{d} \left( \frac{1}{t} \sum_{k=1}^{t} (\vec{g}_k)_i (\vec{g}_k)_j \right) \vec{u}_i \vec{v}_j
\]
\[
    = \delta_{ij} \text{ by 2-independence}
\]
\[
    = \sum_{i=1}^{d} \vec{u}_i \vec{v}_i
\]
\[
    = \vec{u} \cdot \vec{v}.
\]

(2) Using (1),
\[
    \|h(\vec{v})\|^2 = h(\vec{u}) \cdot h(\vec{v}) = \vec{u} \cdot \vec{v} = \|\vec{u}\|^2 = 1.
\]

(3) Using (2),
\[
    \|h^M(\vec{v})\|^2 = \sum_{k} (h^M(\vec{u}))_k^2 \leq \sum_{k} (h(\vec{u}))_k^2 = 1.
\]

(4) From the definition of $h^M(\cdot)$, we have
\[
    \|h(\vec{u}) - h^M(\vec{u})\|^2 = \sum_{k:|h(\vec{u})_k| > \frac{M}{\sqrt{t}}} |h(\vec{u})_k|^2
\]
\[
    = \frac{1}{t} \sum_{k:|\vec{g}_k \cdot \vec{u}| > M} (\vec{g}_k \cdot \vec{u})^2
\]
\[
    \leq \sqrt{\frac{1}{t} \sum_{k:|\vec{g}_k \cdot \vec{u}| > M} (\vec{g}_k \cdot \vec{u})^4} \sqrt{\frac{1}{t} \sum_{k:|\vec{g}_k \cdot \vec{u}| > M} 1}
\]
\[
    \leq \sqrt{3} \sqrt{3} \frac{1}{M^2}
\]
\[
    = \frac{3}{M^2}.
\]
where the second line is by the Cauchy-Schwarz inequality and the third uses

\[ \frac{1}{t} \sum_{k:|\vec{g}_k \cdot \vec{u}| > M} (\vec{g}_k \cdot \vec{u})^4 \leq \frac{1}{t} \sum_k \sum_{i,j,l,m} (\vec{g}_k)_i (\vec{g}_k)_j (\vec{g}_k)_l (\vec{g}_k)_m \vec{u}_i \vec{u}_j \vec{u}_l \vec{u}_m \]
\[ = \sum_i (\vec{u}_i)^4 + 3 \sum_{i \neq l} (\vec{u}_i)^2 (\vec{u}_l)^2 \]
\[ \leq 3(\|\vec{u}\|^2)^2 \]
\[ = 3, \]

where the second line follows by observing that \( \sum_k (\vec{g}_k)_i (\vec{g}_k)_j (\vec{g}_k)_l (\vec{g}_k)_m = 0 \) unless \( i = j \) and \( l = m \), or \( i = l \) and \( j = m \), or \( i = m \) and \( j = l \). From this bound we also get that \( \#\{k:|\vec{g}_k \cdot \vec{u}| > M\} \leq \frac{3}{M^2} t \), as required above.

Remark 9.6. The same guarantees for the rounding procedure can be obtained by taking random projections on Gaussian vectors and doing an analysis based on the Johnson-Lindenstrauss lemma.