Grothendieck's inequality

Theorem (Grothendieck's inequality).
Consider an $m \times n$ matrix $(a_{ij})$ of real numbers. Suppose that for any $x_i, y_j \in \mathbb{R}$

$$\left| \sum_{i,j} a_{ij} x_i y_j \right| \leq \max_i |x_i| \max_j |y_j|$$

(i.e., multiply the $i$th row by $x_i$ & the $j$th column by $y_j$ & sum the entries).

Then for any Hilbert space $H$ and any vectors $u_i, v_j \in H$ we have

$$\left| \sum_{i,j} a_{ij} \langle u_i, v_j \rangle \right| \leq K \max_i \|u_i\| \max_j \|v_j\|.$$ 

$K \leq 1.783$ is an absolute constant.

Remark: while the statement holds for a constant $K \leq 1.783$ we will give a proof that gives $K \leq 8$.

Proof

1) Given an $m \times n$ matrix $A$, let $K = K(A)$ be the smallest $K$ which makes the statement true for every Hilbert space $H$.

Note that $K = \sum_{i,j} |a_{ij}|$ works, so the set of $K$'s that work is not empty.

The key point of the theorem is that $K(A)$ in fact does not depend on $A$ in any.
2) Given $u_i, v_j \in H$, we need to show we can find

$$k \leq \max \|u_i\| \max \|v_j\|.$$ 

Once $u_i, v_j$ are selected, the space $H$ does not play any role any more so we can replace $H$ by its subspace $\tilde{H}$ spanned by all the $u_i$'s and $v_j$'s.

$\tilde{H}$ is a subspace of $\mathbb{R}^n$, so it is isometric with a subspace of $\mathbb{R}^n$. Thus, without loss of generality we can assume $H = \mathbb{R}^n$ with the standard inner product.

3) We need to bound

$$\left| \sum_{i,j} a_{ij} \langle u_i, v_j \rangle \right|.$$

Let's realize $\langle u_i, v_j \rangle$ via random Gaussian vectors.

Let $g \sim \mathcal{N}(0, I_n)$ and define $u_i = \langle g, u_i \rangle$

$$v_j = \langle g, v_j \rangle + g_j.$$ 

$U_i, V_j$ are linear combinations of independent normal Gaussians, so they are mean-zero Gaussians.

Moreover

$$EU_i V_j = E\left( \begin{bmatrix} u_i^T & g_i^T \end{bmatrix} \begin{bmatrix} g_j \end{bmatrix} \right) = u_i^T E g_j g_j^T v_j = u_i^T v_j = \langle u_i, v_j \rangle.$$ 

So

$$\sum_{i,j} a_{ij} \langle u_i, v_j \rangle = E \left( \sum_{i,j} a_{ij} U_i V_j \right).$$
This way we could turn the inner product \(<u_i, v_j>\) into the product \(U_i V_j\) (at the cost of adding the expectation) to which we can apply the assumption of the form:

for a given realization of \(U_i, V_j\) we can use the assumption in the form to write

\[ \sum_{i,j} a_{ij} U_i V_j \leq \max_i |U_i| \max_j |V_j|. \]

The issue here is that \(U_i, V_j\) are normal, so they are not bounded, so can be arbitrarily large.

4) Truncate the RVs \(U_i, V_j\) — separate into two parts, the 1st is bold, the 2nd is unlikely (has small probability).

Given \(R\), let

\[ U_i^- = U_i I\{|U_i| \leq R_{1/4}\} U_i^+ = U_i I\{|U_i| > R_{1/4}\}. \]

Similarly define

\[ V_j^- = V_j I\{|V_j| \leq R_{1/3}\} V_j^+ = V_j I\{|V_j| > R_{1/3}\}. \]

we have \(U_i = U_i^+ + U_i^-\)

\(V_j = V_j^+ + V_j^-\).

We have

\[ \sum a_{ij} U_i V_j = \underbrace{\sum a_{ij} U_i^- V_i^+}_{S_1} + \underbrace{\sum a_{ij} U_i^+ V_i^-}_{S_2} + \underbrace{\sum a_{ij} U_i^- V_i^-}_{S_3} + \underbrace{\sum a_{ij} U_i^+ V_i^+}_{S_4}. \]
For $S_1$ by the hypothesis on the thin 

\[ |S_r| \leq \max_i \| U_i \| \max_j \| V_j \| \leq R^2 \max_i |u_i| \max_j |v_j| \]

so \( |S_r| \leq R^2 \max_i |u_i| \max_j |v_j| \)

5) For $S_2$ we write 

\[ E S_2 = \sum_{i,j} a_{ij} E(U_i^+ V_j^-). \]

Consider $U_i V_j^+$ as elements of the Hilbert space $L_2$ with the inner product 

\[ \langle x, y \rangle_{L_2} = E X Y. \]

Our $k(k)$ works for any Hilbert space so we have 

\[ |E S_2| \leq K \max_i \| U_i^+ \|_{L_2} \max_j \| V_j^- \|_{L_2} \]

Since $U_i = \langle g_i, u_i \rangle$, we have $U_i \sim N(0, \| u \|^2) \sim \| u \| N(0,1)$.

Thus 

\[ \| U_i \|_{L_2}^2 = E U_i^2 \frac{1}{\| u_i \|} \| u_i \| = \| u \|^2 \frac{1}{\| u \|} E(g_i^2 1_{|g_i| > R}) \]

where $g_i \sim N(0,1)$.

A simple integration by parts gives 

\[ \frac{1}{2} B g_i^2 1_{|g_i| > R} = E g_i^2 1_{|g_i| > R} = R \cdot \frac{1}{\| u \|} e^{-R^2/2} + P(|g_i| > R) \]

so 

\[ E g_i^2 1_{|g_i| > R} \leq 2(R + \frac{1}{R}) \frac{1}{\| u \|} e^{-R^2/2} =: C_R \]

we get 

\[ \| U_i \|_{L_2}^2 \leq \| u \|^2 C_R , \| V_j^- \|_{L_2} \leq \| v_j \|^2 \]

We get 

\[ \| U_i \|_{L_2}^2 \leq \| u \|^2 C_R , \| V_j^- \|_{L_2} \leq \| v_j \|^2 \| v_j \| \]
Thus $|E_{S_2}| \leq K \cdot C_R \max \|v_i\| \max \|v_j\|$

Similarly $|E_{S_3}| \leq K \cdot C_R$

$|E_{S_4}| \leq K \cdot C_R^2$

So $|E \sum a_{ij} x_i y_j| \leq (R^2 + K(2C_R + C_R^2))$

So $K$ was the smallest which made $\int$ work for all $H$, so

$K \leq \frac{R^2 + K(2C_R + C_R^2)}{1 - (2C_R + C_R^2)}$ so $K \leq R^2$

Plug in $R = 2.3$, get $K \leq 8$

**Remark:** The hypothesis in the form is that $\forall x, y \in \mathbb{R}$

\[
\left| \sum_{i,j} a_{ij} x_i y_j \right| \leq \max_j |x_i| \max_j |y_j|
\]

This is in fact equivalent to

\[
\left| \sum_{i,j} a_{ij} x_i y_j \right| \leq 1 \quad \forall x_i, y_j \in \{-1, 1\}.
\]

That $\bigcirc \Rightarrow \bigcirc$ is trivial.

Suppose $\bigcirc$ holds. Let $S$ be the subset of $\mathbb{R}^{m \times m}$ consisting of all the vectors $S = \{ (x_1, x_2, \ldots, x_m) \}$ such that

$-1 \leq \sum_{i,j} a_{ij} x_i y_j \leq 1$.

That $\bigcirc$ implies all the vectors $(x_1, x_1, \ldots, x_1) \in S$.

Since $S$ is defined by a collection of linear inequalities, it is convex, so $S$ must contain the convex hull of

$(\pm 1, \ldots, \pm 1)$, so $\forall x_1, x_2, \ldots, x_m \in \mathbb{R}$

\[
\left( \frac{x_1}{\max |x_1|}, \ldots, \frac{x_m}{\max |x_1|} \right) \in S \quad \text{which means} \quad \bigcirc \quad \text{holds.}
\]
Applications of Grotendieck's inequality

In computationally difficult problems often approximate solutions are sought. Grotendieck's inequality can be used to guarantee the approximation will be good. We will look at examples of computationally difficult problems which can be approximated by semi-definite programming, a generalization of linear programming.

Def: A semi-definite program is an optimization problem of the following type:

given n x n matrices $A_1, B_1, \ldots, B_m$, and real numbers $b_1, \ldots, b_m$.

Find an $n \times n$ positive semi-definite matrix $X$ ($X \succeq 0$) which maximizes $\langle A_i, X \rangle$

under the constraints

$X \succeq 0$, $\langle B_i, X \rangle = b_i$ for $i = 1, \ldots, m$.

Note that the inner product of $n \times n$ matrices $A, B$ is

$\langle A, B \rangle = \sum_{ij} a_{ij} b_{ij}$ which can be written as

$\langle A, B \rangle = \text{tr}(A^T B)$.

Rule: The main difference from linear programming is in the constraints: non-negativity in linear programming is replaced by positive semi-definiteness here.

Rule: The set of semi-definite matrices forms a convex set in the space of $n \times n$ matrices (check!)

2. The intersection of that set with the
Constraint hyperplanes \( \langle b_i, x \rangle = b_i \) is still convex, so the semi-definite program is optimizing \( \langle A, x \rangle \) over a convex set, which makes it computationally tractable.

We'll now look at some examples of algorithms which we will approximate by semi-definite programming.

**Problem:** maximize \( \sum_{i,j} A_{ij} x_i x_j \quad x_i \in \{\pm 1\}, \quad i = 1, \ldots, n \)

where \( A \) is a fixed non-symmetric matrix.

Let \( \vec{x} = (x_1, \ldots, x_n) \in \pm 1^n \) be the maximizer &

\( \text{Int}(A) \) be the maximum value achieved by \( \sum_{i,j} A_{ij} x_i x_j \).

The problem is known to be NP-hard.

Instead of solving it, we can find the maximum approximately, up to a constant factor.

Replace the numbers \( x_i = \pm 1 \) by unit vectors \( \vec{x}_i \in \mathbb{R}^n \) &

solve the optimization problem:

\[
\text{maximize } \sum_{i,j=1}^n A_{ij} \langle \vec{x}_i, \vec{x}_j \rangle \quad \text{s.t. } \|\vec{x}_i\|_2 = 1 \quad \forall i.
\]

Let \( X \) be the norm matrix \( X_{ij} = \langle \vec{x}_i, \vec{x}_j \rangle \).

Check \( X \) is positive semi-definite, & every positive semi-definite matrix can be realized in such a way.

We have

\[
\sum_{i,j=1}^n A_{ij} \langle \vec{x}_i, \vec{x}_j \rangle = \sum_{i,j=1}^n A_{ij} X_{ij} = \langle A, x \rangle,
\]

so
\[ \begin{align*}
\text{maximize} & \quad \langle A, x \rangle \\
\text{subject to} & \quad x \geq 0, \quad x_{ki} = 1 + \epsilon_i.
\end{align*} \]

This is a semi-definite program. Let \( \bar{x} \) be the maximizer of \( \text{Spd}(A) \), and the maximum achieved by it.

By setting \( x_i = (x_i, \alpha - \epsilon) \) we see that \( \bar{x} \) cannot do worse than \( x \), so

\[ \text{Int}(A) \leq \text{Spd}(A). \]

On the other hand, if we replace the matrix \( A \) by

\[ \hat{A} = \frac{A}{\text{Int}(A)}, \]

\[ \forall x, \epsilon, \exists i \quad \mid \sum \hat{A}_{ij} x_i x_j \mid \leq 1 \]

so by Grothendieck's inequality

\[ \sum \hat{A}_{ij} \langle x_i, x_j \rangle \leq 2K \quad \forall \mid x_i \mid = 1, \quad i = 1, \ldots, n. \]

where \( K \) is Grothendieck's constant.

(The reason we need \( 2K \) instead of \( K \) is because we are dealing with the symmetric version \( 1 = x_i^T \) of Grothendieck.)

It follows that \( \text{Spd}(A) \leq 2K \text{ Int}(A) \).

So

\[ \text{Int}(A) \leq \text{Spd}(A) \leq 2K \text{ Int}(A) \]

so by solving the semi-definite problem, instead of the original integer optimization problem, we overestimate the maximum by a factor of most \( 2K \approx 3.7 \).
Maximum cuts in graphs

Let $G = (V, E)$ be a graph.

Partition the vertices into 2 disjoint sets $S$ and $\bar{S}$ and count the number of edges between them — this is called a cut.

Let $\text{Max-cut}(G)$ be the maximum possible cut.

Given $G$, computing $\text{Max-cut}(G)$ is NP-hard.

We can use semi-definite programming to approximate it.

To do that, let's phrase the problem in terms of linear algebra.

Number the vertices $1, 2, \ldots, n$ and let $A$ be the adjacency matrix $A_{ij} = \{ 1 \text{ if edge } i \to j \}

A$ is symmetric.

Let $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ indicate the partition of a vertex.

The cut of this partition $\text{Cut}(G, x)$ can be written as

$$
\text{Cut}(G, x) = \frac{1}{2} \sum_{i,j} A_{ij} \frac{1}{x_i + x_j} = \frac{1}{2} \sum_{i,j} A_{ij} \left( \frac{1}{x_i + x_j} \right) = \frac{1}{2} \sum_{i,j} A_{ij} (1 - x_i x_j)
$$

(We divide by 2 to avoid double count.)

So the integer optimization problem is

$$
\text{max-cut}(G) = \frac{1}{4} \max \left\{ \sum_{i,j} A_{ij} (1 - x_i x_j) : x_i \in \{-1, 1\} \forall i \right\}.
$$

Instead, as before, consider the semi-definite program
$$Sdp(G) = \frac{1}{4} \max \left\{ \sum_{i,j=1}^{n} A_{ij} (1 - \langle X_i, X_j \rangle) : X_i \in \mathbb{R}^n, \| X_i \|_2^2 = 1 \forall i \right\}$$

Easy to see that $Sdp(G) \geq \text{Max-cut}(G)$.

Given the optimizer $X = (X_1, -X_2, X_3, -X_4)$ for $Sdp(G)$, we can get a partition for $G$ via a cut $\geq 0.878 Sdp(G)$.

We have vectors $X_1, X_2, X_3, X_4 \in \mathbb{R}^n$.

Choose a random hyperplane through the origin $y$.

Set $y_i = \pm 1$ depending on which side of the hyperplane $x_i$ is.

We can choose the hyperplane by choosing its normal vector uniformly at random from the unit sphere, or choosing $g \sim \mathcal{N}(0, I_n)$ and setting

$$x_i = \text{sign}(\langle g, x_i \rangle).$$

Let $x = (x_1, -x_2, x_3, -x_4)$ be this random partition obtained from rotating $(X_1, -X_2, X_3, -X_4)$. We have

Thus, we have

$$\text{max-cut}(G) \geq E \text{Cut}(G, x) \geq 0.878 Sdp(G) \geq 0.878 \text{Max-cut}(G).$$