SemiDefinite Programming & Randomized Rounding

Fact about positive semidefile symmetric matrices X $(n \times n)$

1. X has nonnegative eigenvalues
2. $X = V^T V$ for some $V \in \mathbb{R}^{m \times n}$ with $m \leq n$
3. $X = \sum_{i=1}^{n} \lambda_i w_i w_i^T$ $\quad w_i \in \mathbb{R}^n$ $\quad \langle w_i, w_i \rangle = \|w_i\|^2 = 1$

$\langle w_i, w_j \rangle = w_i^T w_j = 0$ for $i \neq j$

Most common example of SDP

$$\max\backslash\min \quad \sum_{ij} c_{ij} x_{ij}$$

$$\sum_{ij} a_{ijk} x_{ij} = b_k \qquad \forall k$$

$$x_{ij} = x_{ji} \qquad \forall i, j$$

$$X = (x_{ij}) \succeq 0$$

$\uparrow$ psd condition

This can also be written as

$$\max\backslash\min \quad \sum_{ij} c_{ij} \langle v_i, v_j \rangle$$

$$\sum_{ij} a_{ijk} \langle v_i, v_j \rangle = b_k$$

$$v_i \in \mathbb{R}^n$$

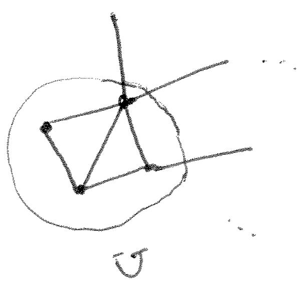
Under some technical conditions SDP can be solved to ε error (additive) in $\text{poly}(n) \log(\frac{1}{\varepsilon})$ time

Why are semidefinite programms useful?

- Gives better approximation algorithms (Max Cut, Correlation Clustering
- Wide variety of applications (network flows etc)
- New tool for your toolbelt

Important Example: (Max-Cut)

Let $G = (V, E)$ with edge weights w_{ij}



$$\max_{U \subset V} c(U, V \setminus U) = \max \sum_{\substack{e:\ \text{one end in } U \\ \text{other in } V \setminus U}} w_e$$

Why not use LP

$$\max \sum w_{ij} x_{ij}$$

s.t

$$x_{ij} \le u_i + u_j \qquad \forall i,j$$

$$x_{ij} \le (1-u_i) + (1-u_j)$$

$$x_{ij}, u_i, u_j \in \{0,1\}$$

$u_i = 1$ if $i \in U$ 0 otherwise

Linear programming relation is bad!

Set $x_i = \frac{1}{2} \ \forall i$, set $x_{ij} = 1 \ \forall (i,j)$

For K_n $w_{ij} = 1$ the resulting opt for the LP $|E| = \binom{n}{2}$,

opt_{IP} int value is $\approx \frac{|E|}{2}$

$\Rightarrow$ LP's cannot give you better apx than 2 (There is easy alg that can give 2 apx greedy, or random

! But semidef programs can do better!

$$\text{(Orig)} \qquad \max \frac{1}{2} \sum_{(i,j) \in E} w_{ij} (1 - y_i y_j)$$

$$y_i \in \{-1, +1\}$$

Cut $U = \{ i \mid y_i = -1 \}$ $W = \{ i \mid y_i = 1 \}$ $W = V \setminus U$

Note if edge (i,j) is in the cut then $y_i y_j = -1$

if not then $y_i y_j = 1$

Consider more general relaxation which is SDP

$$\text{(VP)} \qquad \max \frac{1}{2} \sum_{(i,j) \in E} w_{ij} \left(1 - \langle v_i, v_j \rangle \right)$$

$$\langle v_i, v_i \rangle = 1$$

$$v_i \in \mathbb{R}^n$$

Note $\mathrm{Opt}_{VP} \geq \mathrm{Opt}_{\text{max cut}}$

since any feasible solution to (Orig) denoted y is also feasible for (VP) by setting $v_i = (y_i, 0, \ldots 0)$

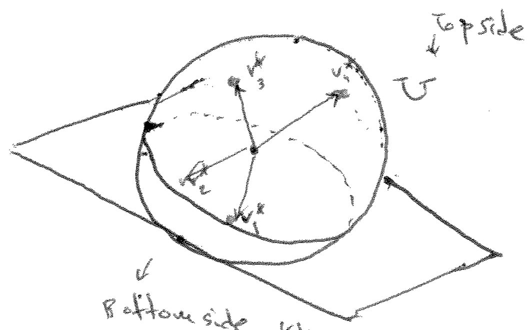
As we mentioned earlier we can solve SDP (under ~~with~~ some condition) to additive ε error for any $\varepsilon > 0$ so we can solve (VP)!

Let $v_1^*, \ldots, v_n^*$ be the opt solution to (VP)

but this is <u>not</u> giving us a cut! (We need to round)

<u>Idea of Goemans Williamson</u> (1994)

$v_1^* \ldots v_n^* \in S^n$ n dim unit sphere.



Randomly (uniformly) cut the sphere in half point that lie on one side of the sphere is the cut

Take $r = (r_1, \ldots, r_n)$

$r_i \sim N(0,1)$

$\frac{r}{\|r\|}$ is uniformly distributed over S^n

$$U = \{ i \mid \langle v_i, r \rangle \geq 0 \}$$

$$W = \{ j \mid \langle v_j, r \rangle < 0 \}$$

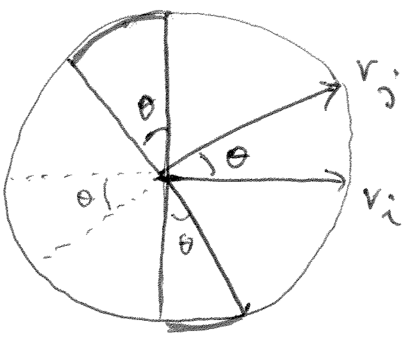
Analysis of the algoritm of Goemans Williamson

We would like to know $Pr[(i,j) \text{ is in the cut}]$

Need two facts

① The projection of r to unit vectors e_1 and e_2 are ind and normally distributed iff e_1 & e_2 are orthogonal

$\Rightarrow$ If r' is projection of r to two dim plane then normalization of $\frac{r'}{\|r'\|}$ is uniformly distributed over the unit circle on that plane.



an place where if r lands (i,j) is in the cut

$$\langle v_i, r\rangle = \langle v_i, r' + r''\rangle = \langle v_i, r'\rangle$$

$$\langle v_j, r\rangle = \langle v_j, r' + r''\rangle = \langle v_j, r'\rangle$$

$$r = r' + r''$$

vector on the unit circle on plane spanned by v_i, v_j

vector orthogonal to plane spanned by v_i, v_j

$$\Rightarrow \quad \theta = \arccos(\langle v_i, v_j\rangle)$$

$$Pr\left[\begin{matrix}(i,j) \text{ is in the} \\ \text{cut}\end{matrix}\right] = \frac{2\theta}{2\pi} = \frac{1}{\pi}\arccos(\langle v_i, v_j\rangle)$$

$$\text{Alg} = \sum_{(i,j)\in E} w_{ij} \Pr\left[(i,j) \text{ in in the cut} \right]$$

$$= \sum_{(i,j)\in E} w_{ij} \frac{1}{\pi} \arccos\left(\langle v_i^*, v_j^* \rangle \right)$$

$$\geq \alpha \sum_{(i,j)\in E} w_{ij} \frac{1}{2} \left(1 - \langle v_i^*, v_j^* \rangle \right)$$

$$\alpha = \min_{-1 \leq x \leq 1} \frac{\frac{1}{\pi} \arccos(x)}{\frac{1}{2}(1-x)} \approx 0.878$$

$$\Rightarrow \frac{1}{\pi} \arccos(x) \geq \alpha \frac{1}{2}(1-x) \quad \forall x \in [-1,1]$$

$$= \cancel{\tfrac{1}{2}} \alpha \, Opt_{vp} \geq \alpha \, Opt_{max\,cut}$$

So our algorith approximates Max cut to factor $\alpha \approx 0.878$ in expectation (wow so much better than $\frac{1}{2}$

- Best known algorithm for Max Cut

If Unique Game Conjecture is true then this is the best possible! unless P=NP