PRINCETON UNIVERSITY
ORFE
Instructor:
Amir Ali Ahmadi

This lecture:

"Approximation algorithms based on convex optimization"

We will cover:

- A 2-approx. alg. for Vertex Cover based on LP (easy and as warmup)
- A .878-approx. alg. for MaxCut based on SDP

(breakthrough result of Goemans and Williamson [GW95])

- Since we know that finding the optimal solution to an NP-hard problem in polynomial time is impossible (unless P=NP), it is natural to ask if we can find (in poly time) a solution whose objective value is guaranteed to be within some multiplicative factor of the optimal value. This is what approx. algs. do.

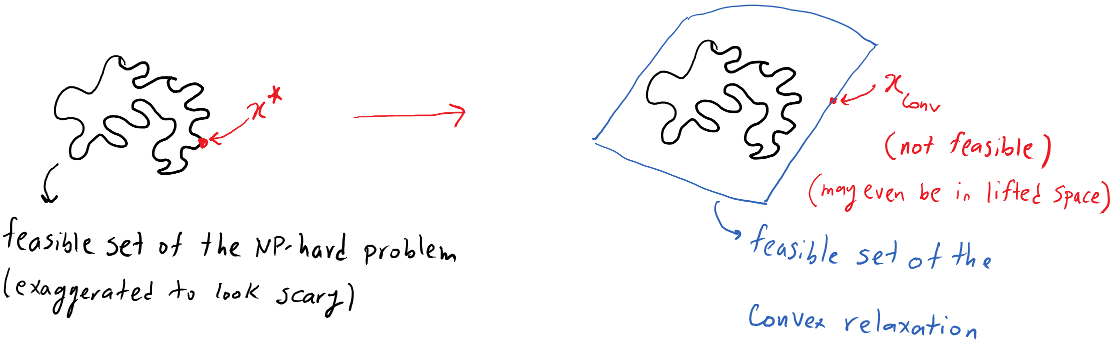
- For a minimization problem with optimal value $f^* \geq 0$, we say that algorithm $\mathcal{A}$ is an α-approximation algorithm, if it runs in polynomial time and produces a solution with objective value $\hat{f}$, such that $f^* \leq \hat{f} \leq \alpha f^*$ (where $\alpha > 1$).

- For a maximization problem with optimal value $f^* \geq 0$, we say that algorithm $\mathcal{A}$ is an α-approximation algorithm, if it runs in polynomial time and produces a solution with objective value $\hat{f}$, such that $\alpha f^* \leq \hat{f} \leq f^*$ (where $0 < \alpha < 1$).

In both cases, we want α to be as close to 1 as possible. In our definitons, we also allow for "randomized algorithms". The bounds then need to hold in expectation.

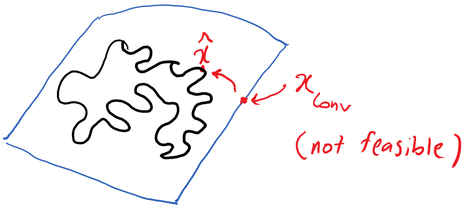
- General outline of convex optimization based approximation algorithms.

Relax:



$f_{conv} := f(x_{conv}) \leq f^* := f(x^*)$ (for a minimization problem)

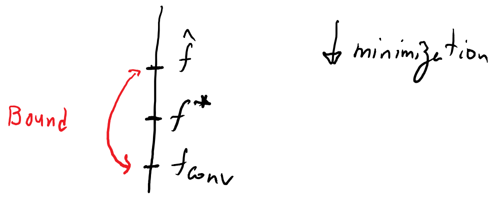
Round:



$\hat{x}$: rounded solution, feasible.

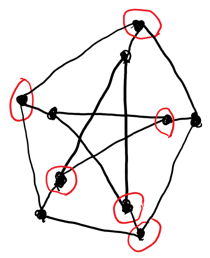
Let $\hat{f} := f(\hat{x})$.

Bound:
- We know $f^* \leq \hat{f}$ (just b/c $\hat{x}$ is feasible).
- Want to bound the gap between f^* and $\hat{f}$, but we have no idea what's f^*.
- But we know $f_{conv} \leq f^*$ $\Rightarrow$ Let's instead bound the gap between f_{conv} and $\hat{f}$. This would also be a valid bound on the ratio of $\hat{f}$ and f^*.



Vertex Cover

Given an undirected unweighted graph $G(V,E)$, find a set of vertices of minimum size that each edge gets touched.



- Valid vertex cover b/c each edge touches at least one red node.
- In fact of minimum size.

- Finding a minimum vertex cover is NP-hard. Here's why:
- Let $n := |V|$, $\alpha(G) :=$ stability number $vc(G) :=$ size of minimum vertex cover.

Then: $vc(G) = n - \alpha(G)$

– why? A set of nodes S is a vertex cover $\Leftrightarrow$ $V \setminus S$ is a stable set

↑ Convince yourself.

- We have already proved that finding $\alpha(G)$ is NP-hard.

Vertex Cover as an integer program:

$$f^* := vc(G) = \min_x \sum_{i=1}^n x_i$$

$$x_i + x_j \geq 1 \quad \forall (i,j) \in E$$

$$x_i \in \{0,1\} \quad i = 1, \ldots, n$$

LP relaxation:

$$f_{LP} := \min \sum_{i=1}^{n} x_i$$
$$x_i + x_j \geq 1 \quad , \quad \text{if } (i,j) \in E$$
$$0 \leq x_i \leq 1 \qquad i = 1, \dots, n$$

Obviously $f_{LP} \leq f^*$. Denote the optimal solution by x_{LP}.

Rounding: Set $\hat{x}_i = \begin{cases} 1, & \text{if } x_{LP,i} \geq 1/2. \\ 0 & \text{otherwise} \end{cases}$

- $\hat{x}$ gives a valid vertex cover b/c $\forall$ edges, one of the two end nodes in the LP solution must be $\geq 1/2$.
- So $f^* \leq \hat{f} := \sum_i \hat{x}_i$

Bounding:

- $\hat{f} \leq 2 f_{LP}$

  b/c in worst case, we are changing a bunch of "$1/2$'s" to "1's".
- $\Rightarrow \hat{f} \leq 2 f^*$

  b/c $f_{LP} \leq f^*$

Overall: $\boxed{f^* \leq \hat{f} \leq 2 f^*}$

This is the best approximation ratio known to date!

$\boxed{\text{Max Cut}}$

Given an undirected graph $G(V,E)$ with nonnegative edge weights w_{ij}, find a partition of the nodes into two disjoint sets V_1 and V_2 ($V_1 \cap V_2 = \phi$, $V_1 \cup V_2 = V$) such that the sum of the edge weights going from V_1 to V_2 is maximized.

- Finding the Max Cut value of a graph is NP-hard (e.g., there's a relatively straightforward reduction from 3SAT).

- Contrast this with Min Cut, which we argued can be solved in poly-time by linear programming.

- We will now produce a (randomized) solution for Max Cut (in poly-time), which in expectation is 87% optimal!

- Denote the Max Cut value of your graph by f^*:

$$f^* = \max_{\text{s.t. } x_i^2=1} \frac{1}{4}\sum_{i,j} w_{ij}(1-x_i x_j) = \frac{1}{4}\sum_{i,j} w_{ij} - \frac{1}{4}\underbrace{\begin{bmatrix} \min & \sum_{i,j} w_{ij} x_i x_j \\ \text{s.t.} & x_i^2 = 1 \end{bmatrix}}_{:= f_2^*}$$

Define a matrix $Q \in S^{n\times n}$ (where $n=|V|$) as $Q_{ij} = \begin{cases} 0 & i=j \\ w_{ij} & i\neq j \end{cases}$

Then, $f_2^* = \min x^T Q x$

s.t. $x_i^2 = 1$.

Here's the standard SDP relaxation for this problem:

$$f_{2SDP} := \min_{X \in S^{n \times n}} \mathrm{Tr}(QX)$$

$X_{ii} = 1$

$X \succeq 0$

(with a constraint $\mathrm{rank}(X)=1$, this would be an equivalent formulation)

Clearly, $f_{2SDP} \leq f_2^*$.

Rounding Step.

- If the optimal solution of the SDP is rank-1, you are happy and you go home.
- If not, take the Cholesky factorization of the optimal solution X:

$$\underset{n\times n}{X} = \underset{n\times r}{V^T} \underset{r\times n}{V}, \quad \text{where } r = \mathrm{rank}(X).$$

- Denote the columns of V by $v_i \in \mathbb{R}^r$: $V = [v_1, \ldots, v_n]$
- Observe that $X_{ij} = v_i^T v_j$
- So $\|v_i\| = 1 \ \forall i$ (b/c $X_{ii} = 1$ must hold).
- So we have n points $v_1, \ldots, v_n$ on the unit sphere S^{r-1} in $\mathbb{R}^r$.
- Generate a point $p \in S^{r-1}$ uniformly at random (e.g., p=randn(r,1); p=p/norm(p,2);)
- Set $x_i = \begin{cases} 1 & \text{if } p^T v_i \geq 0 \\ -1 & \text{if } p^T v_i < 0 \end{cases}$ $\quad i = 1, \ldots, n$.
- That's it.

<u>Bounding</u>:

Consider the hyperplane $\mathcal{P} := \{x \in \mathbb{R}^r \mid p^T x = 0\}$

Let $\hat{f}_2$ denote the expected value of the objective value of our rounded solution:

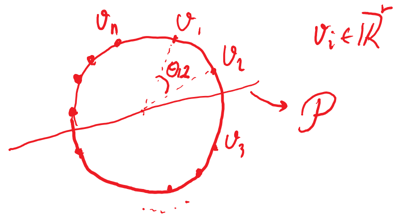
$$\hat{f}_2 = E\left[\sum_{ij} w_{ij} x_i x_j\right] = \sum_{i,j} w_{ij} E[x_i x_j]$$

$$\underset{(i \neq j)}{E[x_i x_j]} = 1 \cdot \Pr[v_i \& v_j \text{ on same side of } \mathcal{P}] - 1 \cdot \underbrace{\Pr[v_i \& v_j \text{ on different sides of } \mathcal{P}]}_{\frac{\theta_{ij}}{\pi} = \frac{1}{\pi} \arccos(v_i^T v_j)}$$

$$= 1 - \frac{\theta_{ij}}{\pi} - \frac{\theta_{ij}}{\pi}$$

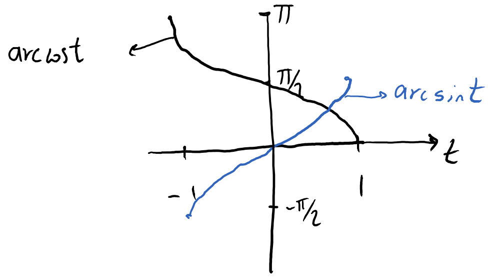
$$= 1 - \frac{2}{\pi} \arccos v_i^T v_j$$

well-defined b/c $X_{ij} \leq 1$ (why?)



$$= \frac{2}{\pi} \arcsin v_i^T v_j$$

$\arcsin t + \arccos t = \pi/2$



$$\Rightarrow \boxed{\hat{f}_2 = \frac{2}{\pi} \sum_{i,j} w_{ij} \arcsin X_{ij}}$$

- Recall that $f^* = \frac{1}{4}\left(\sum_{i,j} w_{ij} - f_2^*\right)$

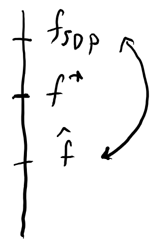
- Let $\hat{f} := \frac{1}{4}\left(\sum_{i,j} w_{ij} - \hat{f}_2\right) = \frac{1}{4}\left(\sum_{i,j} w_{ij} - \frac{2}{\pi}\sum_{i,j} w_{ij} \arcsin X_{ij}\right)$

$$= \frac{1}{4}\sum_{i,j} w_{ij}\left[1 - \frac{2}{\pi}\arcsin X_{ij}\right] = \underline{\underline{\frac{1}{4} \cdot \frac{2}{\pi} \sum_{i,j} w_{ij} \arccos X_{ij}}}$$

We want to relate this to the optimal value of the SDP:

$$f_{SDP} := \frac{1}{4}\left(\sum_{i,j} w_{ij} - f_{2\,SDP}\right)$$

$$= \frac{1}{4}\sum_{ij} w_{ij} - \frac{1}{4}\sum_{i,j} w_{ij} X_{ij} = \underline{\underline{\frac{1}{4}\sum_{i,j} w_{ij}\,(1 - X_{ij})}}$$

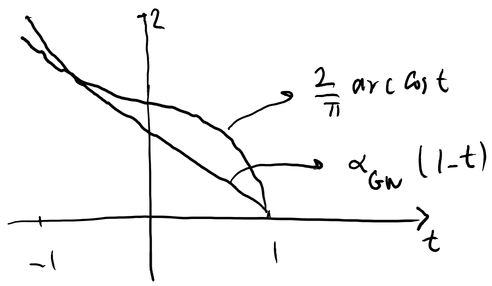


- Want to argue: $\alpha f_{SDP} \leq \hat{f}$

  for α as large as possible.

- We will bound term by term (since $w_{ij} \geq 0$). So we need the largest α for which:

$$\alpha(1-t) \leq \frac{2}{\pi}\arccos t \qquad \forall t \in [0,1]$$



Optimal α: $\alpha_{GW} \approx 0.878$

his car (before the algorithm)

True story!

Notes

Further reading for this lecture can include Chapter 7 of [LV12] and Chapter 3 of [BN01].

References

[GW95] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maxcut and satisfiability problems using semidefinite programming. Journal of the ACM, 1995.

[Pa14] P.A. Parrilo. Lecture notes on Algebraic Techniques and Semidefinite Optimization, MIT, 2014.

[BN01] A. Ben-Tal and A. Nemirovski. Lecture Notes on Modern Convex Optimization. MPS/SIAM Series on Optimization, 2001.

[LV12] M. Laurent and F. Vallentin. Lecture Notes on Semidefinite Optimization 2012.