This lecture:

"Approximation algorithms
based on convex optimization"

We will cover:

0 A 2-approx. alg. for Vertex Cover based on LP (easy and as warmup)
0 A 0.878-approx. alg. for MaxCut based on SDP
(breakthrough result of Goemans and Williamson [GW95])

0 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. alg. do.

0 For a minimization problem with optimal value $f^*$, we say that algorithm $\hat{A}$ is an $\alpha$-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 $\alpha>1$).

0 For a maximization problem with optimal value $f^*$, we say that algorithm $\hat{A}$ is an $\alpha$-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 $\alpha<1$).
In both cases, we want $\lambda$ to be as close to 1 as possible. In our definitions, we also allow for "randomized algorithms": The bounds then need to hold in expectation.

**General outline of convex optimization based approximation algorithms.**

- **Relax:**
  
  Feasible set of the NP-hard problem
  (exaggerated to look scarce)

  \[ f_{\text{conv}} = f(x_{\text{conv}}) \leq f^* = f(x^*) \]  
  (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 $f^*$ is.
  
  But we know
  \[ f_{\text{conv}} \leq f^* \Rightarrow \text{Let's instead bound the gap between } f_{\text{conv}} \text{ and } \hat{f}. \]
  
  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 \( \nu C(G) := \) size of minimum vertex cover.
  - Then: \( \nu C(G) = n - \alpha(G) \)

  - 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:

\[
f^* := \nu C(G) = \min_x \sum_{i=1}^n x_i \quad \text{s.t.} \quad \begin{align*}
    x_i + x_j &\geq 1 \quad \forall (i,j) \in E \\
    x_i &\in \{0,1\} \quad \forall i \in V, \forall \in - n
\end{align*}
\]
LP relaxation:

\[
f_{LP} := \min \sum_{i=1}^{n} x_i
\]

\[x_i \in \{0, \frac{1}{2}\}, \text{ if } (i,j) \in E\]

\[0 \leq x_i \leq 1, \quad i = 1, \ldots, n\]

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

Rounding: Set \(\hat{x}_i = 1\), if \(x_{LP,i} \geq \frac{1}{2}\).

\[\hat{x}_i = 0 \quad \text{otherwise}\]

\(\hat{x}\) gives a valid vertex cover \(\hat{V}\) by \(E\) edges, one of the two end nodes in the LP solution must be \(\geq \frac{1}{2}\).

So \(f^* \geq \hat{f} := \sum_{i} \hat{x}_i\)

Bounding:

\(\hat{f} \leq 2 f_{LP}\)

b/c in worst case, we are changing a bunch of \(\frac{1}{2}'s\) to \(1's\).

\(\Rightarrow \hat{f} \leq 2 f^*\)

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

Overall: \(f^* \leq \hat{f} \leq 2 f^*\)

This is the best approximation ratio known to date!
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 $V_2$ ($V_1 \cap V_2 = \emptyset$, $V_1 \cup V_2 = V$) 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 97% optimal!

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

$$ f^* = \max \frac{1}{4} \sum_{ij} w_{ij} (1 - x_i x_j) = \frac{1}{4} \sum_{ij} w_{ij} - \frac{1}{4} \left[ \min_{ij} \sum_{ij} w_{ij} x_i x_j \right] $$

s.t. $x_i^2 = 1$

Define a matrix $Q \in \mathbb{S}^{nxn}$ (where $n = |V|$) as $Q_{ij} = \begin{cases} 0 & i=j \\ \frac{w_{ij}}{2} & i \neq j \end{cases}$
Then, \( f_2^* = \min x^T Q x \)
\[ \text{s.t.: } x_i^2 = 1. \]

Here's the standard SDP relaxation for this problem:

\[
f_{2,SDP}^* = \min_{X \in S_+^n} \text{Tr}(Q X) \]
\[ X_{ii} = 1 \quad \text{(with a constraint rank}(X) = 1, \text{this would be an equivalent formulation)} \]

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

**Rounding Step.**

1. If the optimal solution of the SDP is rank-1, you are happy and you go home.
2. If not, take the Cholesky factorization of the optimal solution \( X \): \( X = V^T \Sigma V \), where \( r = \text{rank}(X) \).

5. Denote the columns of \( V \) by \( v_i \in \mathbb{R}^n \): \( V = [v_1, \ldots, v_r] \)

6. Observe that \( X_{ij} = v_i^T v_j \)

7. So \( \|v_i\| = 1 \) \( v_i \) (b/c \( X_{ii} = 1 \) must hold).

8. So we have \( n \) points \( v_1, \ldots, v_n \) on the unit sphere \( S^{r-1} \) in \( \mathbb{R}^r \).

9. Generate a point \( p \in S^{r-1} \) uniformly at random (e.g., \( p = \text{randn}(v_i); p = \text{norm}(p,2) \))

10. Set \( X_i = 1 \) if \( p^T v_i > \gamma \), \( i = 1, \ldots, n \).

11. \( X_i = -1 \) if \( p^T v_i < \gamma \).

That's it.
Bounding:

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

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

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

$$\mathbb{E} [x_i x_j] = 1 \cdot \Pr [v_i, v_j \text{ on same side of } \mathcal{P}] - 1 \cdot \Pr [v_i, v_j \text{ on different sides of } \mathcal{P}]$$

$$\leq 1 - \frac{\Theta_{ij}}{\pi} = 1 - \frac{2}{\pi} \arccos (v_i^T v_j)$$

Well-defined

$$\arcsin t + \arccos t = \frac{\pi}{2}$$

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

Recall that $f^* = \frac{1}{4} \left( \sum_{i,j} w_{ij} - \hat{f}_2 \right)$

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

$$= \frac{1}{4} \sum_{i,j} w_{ij} \left[ 1 - \frac{2}{\pi} \arcsin x_{ij} \right] = \frac{1}{4} \cdot \frac{2}{\pi} \sum_{i,j} w_{ij} \arccos x_{ij}$$
We want to relate this to the optimal value of the SDP:

\[ f_{SDP} = \frac{1}{4} \sum_{i,j} (w_{ij} - f_{i,j}) \]

\[ = \frac{1}{4} \sum_{i,j} w_{ij} - \frac{1}{4} \sum_{i,j} w_{ij} x_{ij} = \frac{1}{4} \sum_{i,j} w_{ij} (1 - x_{ij}) \]

\[ \Rightarrow \quad f_{SDP} \leq \hat{f} \]

for \( \alpha \) as large as possible.

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

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

**Optimal** \( \alpha \): \( \alpha_{GW} \approx 0.878 \)
Further reading for this lecture can include Chapter 7 of [LV12] and Chapter 3 of [BN02].


