Randomized rounding of SDPs
- MAX cut
- 3-coloring

max
LP \text{OPT}
OPT

Worst case ratio
\frac{\text{LP OPT}}{\text{OPT}} \text{ called integrality gap.}
**MAXCUT**

**Inputs:** \( G = (V, E) \quad w_{ij} \quad V(i), j \in E \)

**Goals:** partition vertex set so as to max weight of endpoints crossing cut.

**IP formulation of MAXCUT**

\[
\begin{align*}
  x_i &= \begin{cases} 
    0 & \text{on one side of partition} \\
    1 & \text{on other side} 
  \end{cases} \\
  z_{ij} &= \begin{cases} 
    1 & \text{edge } (i,j) \text{ cut} \\
    0 & \text{o.w.} 
  \end{cases} \\
  \text{max} & \quad \sum_{(i,j) \in E} w_{ij} z_{ij} \\
  z_{ij} &\leq x_i + x_j \quad \forall (i,j) \in E \\
  z_{ij} &\leq 2 - (x_i + x_j) \quad \forall (i,j) \in E \\
  x_i &\in \{0, 1\} \quad \forall i \in V \\
  z_{ij} &\in \{0, 1\} \quad \forall (i,j) \in E
\end{align*}
\]

\* no polynomial sized LP relaxation of MAXCUT has integrality gap \( > \frac{1}{2} \).
Another approach:

First, notation change:

\[ \forall i: \quad x_i \in \{-1,1\} \]

define \[ y_{ij} = x_i x_j \quad \forall i,j \in V \]

\[
\max \sum_{(i,j) \in E} w_{ij} \frac{1}{2} [x_i + x_j]
\]

Exactly captures \text{MAX-CUT}!

\[
\max \sum_{(i,j) \in E} \frac{1}{2} (1 - y_{ij})
\]

\[
y_{ij} = y_{ji} \quad \forall i,j \in V
\]

\[
y_i = 1 \quad \forall i \in V
\]

\text{Idea: enforce} \quad \text{brown by adding linear inequalities to purple.}
Intro to semi-definite programming

Linear programming where vars are entries in psd matrix

**Defn** If $A$ is a symmetric $n \times n$ matrix then $A$ is a positive semi-definite (psd) matrix if any of the following equivalent conditions hold:

1. $V \in \mathbb{R}^n$, $VA \geq 0$
2. $A$ has nonnegative eigenvalues
3. $A = VTV$ for some $m \times n$ matrix $V$, $m \geq n$
4. $A = \sum_{i=1}^{n} \lambda_i x_i x_i^T$ for some $\lambda_i \geq 0$ and orthonormal vectors $x_i \in \mathbb{R}^n$
Intro to semi-definite programming

Linear programming where vars are entries in PSD matrix

**Defn**
If $A$ is a symmetric $n$ by $n$ matrix then $A$ is a positive semi-definite (PSD) matrix if $A \succeq 0$.

Any of the following equivalent conditions hold:
1. $V \in \mathbb{R}^n$, $c^T A c \geq 0$
2. $A$ has nonnegative eigenvalues
3. $A = V^T V$ for some $m \times n$ matrix $V$, $m \leq n$
4. $A = \sum_{i=1}^n \lambda_i x_i x_i^T$ for some $\lambda_i \geq 0$ and orthonormal vectors $x_i \in \mathbb{R}^n$

**Semidefinite program (SDP)**

max or min $\sum_{i,j} c_{ij} x_{ij}$

subject to $\sum_{i,j} a_{ijk} x_{ij} = b_k$

$x_{ij} \geq 0$, $V_{i,j}$

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

$= \text{Vector program}$

max or min $\sum_{i,j} c_{ij} (v_i \cdot v_j)$

subject to $\sum_{i,j} a_{ijk} (v_i \cdot v_j) = b_k$

$v_i \in \mathbb{R}^n$, $i=1, \ldots, n$

given $X \Rightarrow x_{ij} = \langle v_i, v_j \rangle$

set $v_i$ to be $i$th column of $V$

**Key facts**

SDPs can be solved to within additive error $\epsilon$ in time $\text{poly}(\text{size of input}, \log(\frac{1}{\epsilon}))$

in our discussions, we ignore additive error $\epsilon$
Recap:

1. Opt solution to brown + purple = Opt of MaxCut

2. Brown \Rightarrow

\[
\text{max } \sum_{(i,j) \in E} \frac{1}{2}(1-y_{ij})
\]

\[
y_{ij} = y_{ji} \quad \forall ij \in V
\]

\[
y_{ii} = 1 \quad \forall i \in V
\]

These constraints \( y_{ii} = 0 \) in \( V \times \mathbb{R}^n \)

\[
\text{Yields a semidefinite programming relaxation of MaxCut}
\]

3. \( \text{Yields a semidefinite programming relaxation of MaxCut} \)

\[
\text{can be solved efficiently}
\]

\[
\text{using the ellipsoid alg.}
\]

\[
\text{up to error } \epsilon
\]

We can solve this, "round" results \( \Rightarrow \) int soln

\[
\Rightarrow \text{prove that it gives}
\]

\[
\text{pretty good approx.}
\]
Can equivalently write SDP relaxation as a vector program

\[
\max \sum_{(i,j) \in E} \frac{1}{2} (1 - y_{ij}) \\
y_{ij} = y_{ji} \quad \forall i,j \in V \\
y_{ii} = 1 \quad \forall i \in V \\
y = (y_1, \ldots, y_n)
\]

Relaxation check:
\[
\text{OPT} \leq \text{OPT}_{\text{SDP}}(G)
\]

\[
\max \sum_{(i,j) \in E} \frac{1}{2} (1 - v_i \cdot v_j) \\
v_i \cdot v_i = 1
\]

i.e. \(v_i\)'s are unit vectors \(v_i \in \mathbb{R}^n\)
Example

Maximize \( \sum_{(i,j) \in E} w_{ij} \frac{1}{2} (1 - \cos(\text{angle} (v_i, v_j))) \)

\[ \leq 0.9 \]

all weights equal

\( \text{OPT} = 4 \)
\( \text{SDP OPT} \approx 4.5 \)

ratio \( \geq \frac{4}{4.5} \approx 0.89 \)

Lovász Umbrella

all edges at angle \( \frac{4\pi}{5} \) (144°)

Lovász umbrella

all edges at angle \( \frac{4\pi}{5} \)

\( \cos \left( \frac{4\pi}{5} \right) = -\frac{1}{2} \approx -0.8 \)

golden ratio

\( 1 + \frac{\sqrt{5}}{2} \)
**MAXCUT**

Inputs $G = (V, E)$, $w_{ij}$, $V(1) \subseteq E$

Goal: partition vertex set so as to max weight of endpoints crossing cut.

Vector programming relaxation

$$\max \frac{1}{2} \sum_{i \neq j} w_{ij}(1 - \mathbf{v}_i \cdot \mathbf{v}_j)$$

$$\mathbf{v}_i \cdot \mathbf{v}_j = 1 \quad \forall v \in V$$

$$\mathbf{v}_i \in \mathbb{R}^n$$

Can solve SDP in poly time.

**Claim** $\text{MAXCUT OPT} \leq \text{SDP OPT}$

But how to round? get large contribution to $\text{OPT}$ when $v_i, v_j$ very far.

**Random hyperplane rounding**

Solve SDP $\rightarrow v_1^*, v_2^*, \ldots, v_n^*$

pick random hyperplane thru origin

partition vertices based on which side of hyperplane
1. If there is an approx alg for MAXCUT with approx ratio \( \geq 0.941 \), then \( P = NP \).

2. If the "unique games conjecture" is true, there is no approx alg for MAXCUT with approx ratio better than 0.878.

3. Int gap of the [GW] SDP = 0.878...

4. Every poly sized LP relaxation of MaxCut has integrality gap at most 1.5...
3-Coloring a 3-colorable graph

Given graph $G = (V, E)$ & promise that it is 3-colorable.

What is $\min k$ s.t. we can find a $k$-coloring of $G$ in poly time?

Simple results:

1. A graph with max degree $\Delta$ can be colored with $\leq \Delta + 1$ colors.

2. A 3-colorable graph can be colored with $O(\sqrt{n})$ colors.

Find a vertex $v$ of deg $\geq \sqrt{n}$

Use 3 colors to color $v$ & its neighbors (neighborhood 2-colorable)

Remove $v$ & its neighbors from graph

An SDP-based alg

$$\min \begin{array}{ll}
\text{st.} & \sum v_i v_j \leq \lambda \\
& \sum v_i = 1 \\
& v_i = 1 \\
& v_i \in \mathbb{R}^n
\end{array}$$

Claim:

if graph is 3-colorable

$\lambda \leq -\frac{1}{2}$

$\cos(\frac{\pi}{3}) = -\frac{1}{2}$
\[ \min \lambda \]
\[ \text{st.} \quad \nu_i \cdot \nu_j \leq \lambda \quad \forall (i, j) \in E \]
\[ \nu_i \cdot \nu_i = 1 \quad \forall i \]
\[ \nu_i \in \mathbb{R}^n \quad \forall i \]

Claims

if graph is 3-colorable

\[ \lambda \leq -\frac{1}{2} \]

Aside: If \( G \) has a triangle, then

optimal soln to SDP has \( \lambda^* \geq \frac{1}{2} \)

Proof: Suppose

\[ 0 \leq \left( \nu_1^2 + \nu_2^2 + \nu_3^2, \nu_1 \nu_2 + \nu_2 \nu_3 + \nu_3 \nu_1 \right) = \nu_1 \nu_1 + \nu_2 \nu_2 + \nu_3 \nu_3 + \nu_1 \nu_2 + \nu_2 \nu_3 + \nu_3 \nu_1 \]

Algorithm

1. Solve SDP (\( \ast \)) \( \Rightarrow \nu_i^* \quad i = 1, \ldots, n \)
2. Choose \( t \) random hyperplanes thru origin
3. Color vertices in each region w/diff color
4. remove any edges properly colored
5. Repeat steps 2-4 until have proper coloring
One execution of step 2 uses at colors.

**Goal:** produce semi-coloring w.p. $\geq \frac{1}{2}$ \((\star)\)

- coloring of nodes s.t.,
- $\leq \frac{n}{4}$ edges have same color at both endpoints
  $\implies$ at least $\frac{n}{2}$ vertices properly colored.

**Observation:** $k$ colors sufficient to get semi-coloring,
$\implies$ graph can be properly colored with $O(k \log n)$ colors

What should $t$ be to guarantee \((\star)\)?

Fix $(i,j) \in E$

Pr($i \& j$ get same color)

$\implies E(\# \text{edges with same color})$
Let $\Delta^*$ be a parameter

1. Pick a vertex of deg $\geq \Delta^*$ & 3-color its 3-neighbors \[
\left\{ \begin{array}{c}
\leq 3 \frac{n}{\Delta^*} \text{ colors}
\end{array} \right.
\]

2. Repeat step 1 until all vertices have degree $\leq \Delta^*$

3. Run SDP-based alg to color rest \[
\tilde{O}(\Delta^*^{1.32}) \text{ colors}
\]

Choose $\Delta^*$ to minimize \[
\frac{3n}{\Delta^*} + (\Delta^*)^{10.92}
\]

\[\Rightarrow \Delta^* = n^{10.92} \Rightarrow \tilde{O}(n^{0.39})\]

Current best: $O(n^{0.199})$

NP-hard to color with 4 colors

Huge open problem: Is there an alg for 3-coloring a 3-colorable graph that uses polylog $n$ colors?

Next time: will use linear programming duality

- lower bounds on randomized tiling gaps
- design randomized alg for online problems