AM 221: Advanced Optimization Spring 2018

Rasmus Kyng *Lecture 20 — Monday, April 9th*

1 Overview

In particular we will study the Max-Cut problem which is an example of non-monotone submodular maximization.

We will learn about a local search based approximation algorithm for maximum cut. Next, we will start familiarizing ourselves with Semi-Definite Programming, in preparation for the next lecture, where we will see a rounding algorithm for Max Cut based on semi-definite programming. This famous algorithm due to Goemans and Williamson achieves a better approximation ratio than the one of the local search algorithm.

2 The Max-Cut problem

Definition 1. *Given an undirected graph $G = (V, E)$, the* ***cut induced by*** $S \subseteq V$ *in* G *is the set of edges "leaving"* S*:*

$$C(S) = |\{\{u, v\} \mid u \in S, v \notin S, \{u, v\} \in E\}|$$

The Max-Cut problem is then the following:

- **Input.** Undirected graph $G = (V, E)$.
- **Goal.** Find S, such that $C(S)$ is maximal.

Max-Cut is known to be NP-hard so we will focus on designing approximation algorithms.

2.1 Local search algorithm

Consider the algorithm described in Algorithm 1. An illustration of this algorithm is given in Figure 1.

It is not clear a priori that this algorithm terminates because the algorithm could keep finding improvements forever. This is in contrast to the greedy algorithm where the budget was limiting the number of steps. However this is not the case as we now prove.

Proposition 2. *Algorithm 1 terminates in at most $m = |E|$ steps.*

Proof. At every step of this algorithm the value of $C(S)$ either increases by one or not at all. When $C(S)$ no longer increases, this means that the algorithm cannot find an improvement and terminates.

Algorithm 1 Local search algorithm for Max-Cut

```
1: $S \leftarrow$ arbitrary set of nodes
2: if $\exists v \in S,\ c(S \setminus v) > C(S)$ then
3:     $S \leftarrow S \setminus v$
4: end if
5: if $\exists v \notin S,\ c(S \cup v) > C(S)$ then
6:     $S \leftarrow S \cup v$
7: end if
8: repeat lines 2 to 7 until there is no improvement.
9: return $S$
```

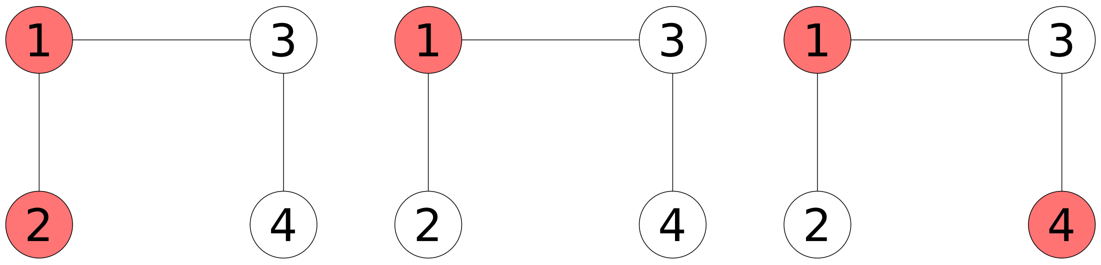

Figure 1: An illustration of the local search algorithm for Max-Cut. The algorithm starts with set $S = \{1, 2\}$ which has value $C(S) = 1$. Then removing 2 improves the value $C(S)$ to 2. Finally adding 4 improves the value to $C(S) = 3$ which is optimal for this graph.

Since $|E|$ is an obvious upper-bound on the optimal value of $C(S)$ this implies that the number of steps is at most m. □

Theorem 3. *Algorithm 1 is a 1/2-approximation algorithm for the Max-Cut problem.*

Proof. Let us define $C(v, T)$ the number of edges from $v \in V$ to $T \subseteq V \setminus \{v\}$:

$$C(v, T) = |\{\{u, v\} \mid u \in T, \{u, v\} \in E\}|$$

Using this definition, we can rewrite $C(S)$:

$$C(S) = \sum_{v \in S} C(v, V \setminus S)$$

we could equivalently write:

$$C(S) = \sum_{u \notin S} C(u, S)$$

or even better, we can combine both expressions:

$$C(S) = \frac{1}{2}\left(\sum_{u \notin S} C(u, S) + \sum_{v \in S} C(v, V \setminus S)\right)$$

Let us write $d(v)$ the degree of $v \in V$. The first observation is that:

$$\forall v \in S,\ C(v, V \setminus S) \geq \frac{d(v)}{2}$$

Indeed, if this was not the case, removing v from S would increase the value of the cut. Similarly we have:

$$\forall u \notin S,\ C(u,S) \geq \frac{d(u)}{2}$$

otherwise adding u to S would increase the value of the cut. Plugging this in the above expression gives:

$$\begin{aligned} C(S) &\geq \frac{1}{2}\left(\sum_{u\notin S}\frac{d(u)}{2} + \sum_{v\in S}\frac{d(v)}{2}\right) \\ &= \frac{1}{4}\sum_{v\in V} d(v) = \frac{2m}{4} = \frac{m}{2} \end{aligned}$$

We can now conclude using that the optimal cut has value upper-bounded by $|E| \leq m$. □

2.2 Randomized Algorithm

We will now consider a simple randomized algorithm.

Algorithm 2 Randomized algorithm for Max-Cut

```
1: $S \leftarrow \emptyset$
2: for $u \in S$ do
3:     add $u$ to $S$ with probability $\frac{1}{2}$
4: end for
5: return $S$
```

Theorem 4. *Let S be the set returned by Algorithm 2. Then $\mathbb{E}[C(S)] = \frac{m}{2}$.*

This theorem again implies that Algorithm 2 is a 1/2-approximation algorithm.

Proof. Define for each edge $e \in E$ the following random variable:

$$X_e = \begin{cases} 1 & \text{if } e \text{ is a cut edge} \\ 0 & \text{otherwise} \end{cases}$$

We have $\mathbb{P}[X_e = 1] = \frac{1}{2}$. Indeed, writing $e = (u,v)$, e is an edge cut if $u \in S$ and $v \notin S$ which occurs with probability $\frac{1}{4}$ or if $u \notin S$ and $v \in S$ which also occurs with probability $\frac{1}{4}$. By linearity of expectation:

$$\mathbb{E}[C(S)] = \mathbb{E}\left[\sum_{e\in E} X_e\right] = \sum_{e\in E}\mathbb{E}[X_e] = \sum_{e\in E}\mathbb{P}[X_e = 1] = \frac{m}{2}$$ □

2.3 Generalization to non-monotone submodular maximization

Max-Cut is in fact an example of non-monotone submodular maximization problem. Recall that a function f is said to be *monotone* iff $S \subseteq T \Rightarrow f(S) \leq f(T)$.

Definition 5. *A function f is called* ***symmetric*** *if $f(S) = f(N \setminus S)$.*

Observe that the cut function in the Max-Cut problem is symmetric.

Theorem 6. *For any non-monotone subdmodular function, the algorithm which selects each element independently with probability $1/2$ is a $1/4$ approximation algorithm in expectation. If the function is symmetric then the approximation ratio is $1/2$.*

Historical note. It was believed for a long time that $\frac{1}{2}$ was the best achievable ratio for symmetric submodular functions until the famous result by Goemans and Williamson in 1994 which gives an SDP-based algorithm for Max-Cut achieving an approximation ratio of 0.878.

3 Introduction to Semi-Definite Programming

In the previous lecture, we saw a rounding algorithm for Set Cover based on Linear Programming. In the next lecture, we will see a rounding algorithm for Maximum Cut based on Semi-Definite Programming. To prepare for this, we end this lecture with a review of some linear algebra and an introduction to semi-definite programming.

3.1 Linear Algebra Review

Let $S_n \subset \mathbb{R}^n$ denote the set of symmetric real $n \times n$ matrices, i.e. for all $A \in S_n$, we have $A = A^\intercal$.

Let $S_n^+ \subset S_n$ denote the set of symmetric real $n \times n$ matrices with non-negative eigenvalues.

Theorem 7. *(Spectral Theorem for Symmetric Matrices) Every symmetric matrix $A \in S_n$, A has n eigenvalues, $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$, with corresponding eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots \mathbf{u}_n$. Let the eigenvectors be scaled to each have unit norm. Let*

$$U = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \ldots & \mathbf{u}_n \end{bmatrix}.$$

Then $U^\intercal U = I$, i.e. the eigenvectors form an orthonormal basis. Finally, $A = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^\intercal$

Definition 8. *Recall we say a matrix $A \in S_n$ is Positive Semi-Definite (PSD) when*

$$\forall \mathbf{x} \in \mathbb{R}^n . \mathbf{x}^\intercal A \mathbf{x} \geq 0.$$

Also use the notation $A \succeq 0$ (or $0 \preceq A$) to denote that A is PSD. Finally, for two symmetric matrices $A, B \in S_n$, we say $A \preceq B$ if and only if $0 \preceq B - A$.

Fact 9. *The $A \preceq B$ defines a partial order on symmetric matrices.*

Theorem 10. *(Courant's Minimax Principle) For $A \in S_n$,*

$$\lambda_k = \min_{\substack{U \subseteq \mathbb{R}^n \\ \text{s.t. } U \text{ is a} \\ \text{subspace of dim } k}} \max_{\substack{\mathbf{x} \in U \\ ||\mathbf{x}||^2 = 1}} \mathbf{x}^\intercal A \mathbf{x}$$

and

$$\lambda_k = \max_{\substack{U \subseteq \mathbb{R}^n \\ \text{s.t. } U \text{ is a} \\ \text{subspace of dim } n+1-k}} \min_{\substack{\mathbf{x} \in U \\ ||\mathbf{x}||^2 = 1}} \mathbf{x}^\intercal A \mathbf{x}$$

Fact 11. $S_n^+ = \{A : A \succeq 0\}$.

Proof. Note that by Courant's Minimax Principle $\lambda_{\min} = \lambda_1 = \min_{\substack{\mathbf{x}\in\mathbb{R}^n \\ ||\mathbf{x}||^2=1}} \mathbf{x}^\intercal A\mathbf{x}$, so the smallest eigenvalue is non-negative if and only if the minimum of the quadratic form in non-negative. $\square$

Fact 12. *S_n^+ is a convex cone.*

Proof. Suppose $A, B \succeq 0$, then for $\alpha, \beta \geq 0$, we have

$$\forall \mathbf{x} \in \mathbb{R}^n . \mathbf{x}^\intercal(\alpha A + \beta B)\mathbf{x} = \alpha \mathbf{x}^\intercal A\mathbf{x} + \beta \mathbf{x}^\intercal B\mathbf{x} \geq 0.$$

$\square$

Definition 13. *(Frobenius Dot Product) For $X, Y \in \mathbb{R}^{n\times n}$, we define $X \bullet Y = \operatorname{Tr}(X^\intercal Y) = \sum_{i,j} X_{ij}Y_{ij}$.*

Note that if we treat the two matrices as vectors of length n^2, this is just the usual dot product.

3.2 Semi-Definite Programs

In general, we can write a Linear Program on n^2 variables as

$$\max_{\substack{X\subseteq\mathbb{R}^{n\times n} \\ \forall i\in[m].A_i\bullet X\geq b_i.}} \quad C \bullet X$$

The optimization program we get by adding an additional constraint $X \succeq 0$ is referred to as a Semi-Definite Program or SDP.

$$\max_{\substack{X\subseteq\mathbb{R}^{n\times n} \\ \forall i\in[m].A_i\bullet X\geq b_i. \\ X\succeq 0}} \quad C \bullet X$$

Theorem 14. *If the Semi-Definite Program on n^2 variables is written using L bits in total, it can be solved to ϵ additive error in time* $\operatorname{poly}(n, L, \log(1/\epsilon))$.

The algorithms to solving SDPs are based on Interior Point Methods.

3.3 Examples of SDPs

Note for 2×2 matrices, $X = \begin{bmatrix} x & z \\ z & y \end{bmatrix} \Leftrightarrow x \geq 0, y \geq 0, xy \geq z^2$. The feasible set for these constraints is shown in Figure 2.

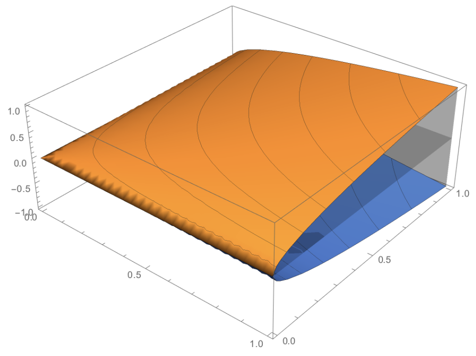

Figure 2: x, y, z plot showing feasible region (shaded gray, between the orange and blue surfaces.) for the constraints $x \geq 0, y \geq 0, xy \geq z^2$.

Noting that $x + y = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \bullet \begin{bmatrix} x & z \\ z & y \end{bmatrix}$ and $z = \begin{pmatrix} 0 & 1/2 \\ 1/2 & 0 \end{pmatrix} \bullet \begin{bmatrix} x & z \\ z & y \end{bmatrix}$, we can write the following program as an SDP:

$$\max_{\substack{x,y,z \subseteq \mathbb{R}^3 \\ z \geq 1 \\ x \geq 0, y \geq 0, xy \geq z^2}} \quad x + y$$

The optimal value of this program will be 2, at the point $(x, y, z) = (1, 1, 1)$. We can prove a matching lower bound using the AM-GM inequality.

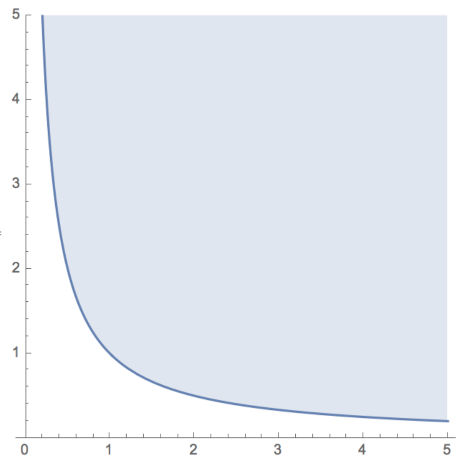

Figure 3: x, y plot showing feasible region (shaded blue) for the constraints $x \geq 0, y \geq 0, xy \geq 1$.

In fact, algorithms for SDPs can also be extended to handle other objectives, such as $\log\det(X^{-1})$, which you proved is convex over PSD matrices in HW7.

There, we considered a program

$$\begin{aligned} \min_{W \in \mathbf{S}_d^{++}} \quad & \log\det(W^{-1}) \\ \text{s.t. } & \mathbf{x}_i^\intercal W \mathbf{x}_i \leq 1, \ 1 \leq i \leq n \end{aligned} \tag{1}$$

Note that the constraint $\mathbf{x}_i^\intercal W \mathbf{x}_i \le 1$ can be written as $W \bullet (\mathbf{x}_i \mathbf{x}_i^\intercal) \le 1$, so we can also formulate this as an SDP, although with a non-linear objective.