ORIE 6334 Bridging Continuous and Discrete Optimization November 3, 2019

Lecture 18

Lecturer: David P. Williamson *Scribe: Dylan Tsai*

1 Spectral Sparsifiers

Today we will introduce the notion of spectral sparsifiers and an algorithm to construct a spectral sparsifier by randomly sampling edges using their effective resistance.

Definition 1 *Let $G = (V, E)$ be an unweighted graph and $H = (V, E')$ be a graph with weights $w(i, j) \geq 0$ $\forall (i, j) \in E'$. Then H is a spectral sparsifier of G if*

$$(1 - \varepsilon)L_G \preceq L_H \preceq (1 + \varepsilon)L_G.$$

We can find a spectral sparsifier H of G such that $|E'| = O((n \log n)/\varepsilon^2)$.

Note that the notion of spectral sparsifier is a strengthening of a cut sparsifier.

Definition 2 *H is a cut sparsifier of G if $\forall S \subseteq V$,*

$$(1 - \varepsilon)|\delta_G(S)| \leq w(\delta_H(S)) \leq (1 + \varepsilon)|\delta_G(S)|,$$

where $\delta_G(S)$ is the set of edges in E with exactly one endpoint in S, $\delta_H(S)$ is the set of edges in E' with exactly one endpoint in S, and $w(\delta_H(S)) = \sum_{(i,j)\in\delta_H(S)} w(i, j)$.

Why is a spectral sparsifier a stronger notion? Given some $S \subseteq V$, let

$$x(i) = \begin{cases} 1, & \text{if } i \in S \\ 0, & \text{otherwise.} \end{cases}$$

Then

$$x^T L_G x = \sum_{(i,j)\in E} (x(i) - x(j))^2 = |\delta_G(S)|$$

$$x^T L_H x = \sum_{(i,j)\in E} w(i, j)(x(i) - x(j))^2 = w(\delta_H(S)).$$

Then $L_H \preceq (1 + \varepsilon)L_G$,

$$\iff (1 + \varepsilon)L_G - L_H \succeq 0 \Rightarrow (1 + \varepsilon)x^T L_G x - x^T L_H x \geq 0 \Rightarrow (1 + \varepsilon)|\delta_G(S)| \geq w(\delta_H(S)).$$

Similarly, if $L_H \succeq (1 - \varepsilon)L_G$, we can show that $w(\delta_H(S)) \geq (1 - \varepsilon)|\delta_G(S)|$.

We use cut sparsifiers in algorithms that find cuts (sparse cuts, min s-t cuts) in order to replace m in the runtime with $O((n \log n)/\varepsilon^2)$. Of course, this is only useful if we can find the cut sparsifier in the first place in time that's at most the time to run the cut algorithm.

[0]This lecture is based in part on the paper by Spielman and Srivastava from 2011, `http://epubs.siam.org/doi/abs/10.1137/080734029`; a lecture by Harvey and Cargèse from 2015, `http://www.cs.ubc.ca/~nickhar/Cargese3.pdf`; and a lecture by Lau from 2015, `https://cs.uwaterloo.ca/~lapchi/cs798/notes/L17.pdf`.

2 Algorithm for Graph Sparsification

The basic idea behind this algorithm is to randomly sample edges and then use Chernoff bounds to bound the probabilities that a sample is far away from the mean.

However, we have to be careful about how we sample. For instance, we can't sample edges with uniform probability. To illustrate, consider a graph with two dense subgraphs and one edge connecting them. If we sample with uniform probability, we get a good estimate of a cut through one of the dense regions. But we get a poor estimate of the cut between the regions, because it is completely dependent on whether we select the one connecting edge.

The main idea used (starting with Benczur, Karger 1996, and nicely codified by Fung, Hariharan, Harvey, Panigraphi 2011) is to sample (i, j) with probability $\propto 1/\lambda(i, j)$, where $\lambda(i, j)$ is a lower bound on the size of an i-j cut.

We can set $\lambda(i, j)$ to be the size of the minimum i-j cut; however, recall that spectral sparsifiers are used to speed up cut algorithms, so needing to find a min cut for all edges (i, j) would defeat the purpose of the spectral sparsifier. The algorithm we will consider today is from Spielman, Srivastava 2011. To construct a spectral sparsifier, we sample (i, j) with probability $\propto r_{\text{eff}}(i, j)$, the effective resistances. We will show later that we can compute these r_{eff} efficiently.

Algorithm 1: Graph Sparsification by Effective Resistances

```
$w(i,j) \leftarrow 0 \ \forall (i,j) \in E$
Compute $r_{\text{eff}}(i,j) \ \forall (i,j) \in E$
$\ell \leftarrow (6(n-1)\ln n)/\varepsilon^2$
for $k \leftarrow 1$ to $\ell$ do
    Pick edge $(i,j)$ with probability $r_{\text{eff}}(i,j)/(n-1)$
    $w(i,j) \leftarrow w(i,j) + \frac{n-1}{\ell \cdot r_{\text{eff}}(i,j)}$
end
$E' \leftarrow \{(i,j) \in E : w(i,j) > 0\}$
```

We first need the following theorem, which we do not prove.

Theorem 1 (Foster's Theorem)

$$\sum_{(i,j)\in E} \frac{r_{\textit{eff}}(i,j)}{r(i,j)} = n - 1.$$

As a result, we see that $\frac{r_{\text{eff}}(i,j)}{(n-1)}$ is a probability distribution on E.

Observation 1 $E' = \{(i,j) \in E : w(i,j) > 0\}$, *and so* $|E'| \leq \ell = O(n \log n/\epsilon^2)$

Observation 2 *The expected total weight of the edges is m.*

As only ℓ changes and the probability cancels out the other terms, the expected value increases by $1/\ell$ per edge at each iteration. As there are ℓ iterations, the expected value of $(n-1)/(\ell \cdot r_{\text{eff}}(i,j)) = 1$. Then summing over m edges, the expected total weight is m.

We now state the Matrix Chernoff bound (a proof can be found in the 2016 notes).

Theorem 2 (Tropp 2011) *Let $X_1, \ldots, X_\ell$ be independent, random, symmetric $n \times n$ matrices such that $0 \preceq X_k \preceq R \cdot I$ for a scalar R. Let $\mu_{min} \cdot I \preceq \sum_{k=1}^{\ell} \mathbf{E}[X_k] \preceq \mu_{max} \cdot I$. Then for all $0 \le \delta < 1$,*

$$\mathbf{Pr}\Big[\lambda_{max}\Big(\sum_{k=1}^{\ell} X_k\Big) \ge (1+\delta)\mu_{max}\Big] \le n \cdot \exp\Big(-\delta^2 \frac{\mu_{max}}{3R}\Big),$$

$$\mathbf{Pr}\Big[\lambda_{min}\Big(\sum_{k+1}^{\ell} X_k\Big) \le (1-\delta)\mu_{min}\Big] \le n \cdot \exp\Big(-\delta^2 \frac{\mu_{min}}{2R}\Big).$$

How do we translate the proof that H is a spectral sparsifier into bounds on the minimum and maximum eigenvalue of a matrix? The following result is what allows us to do so.

Lemma 3 *$L_H \preceq (1+\varepsilon)L_G$ iff $L_G^{\dagger/2} L_H L_G^{\dagger/2} \preceq (1+\varepsilon)I$ iff $\lambda_{max}(L_G^{\dagger/2} L_H L_G^{\dagger/2}) \le (1+\varepsilon)$, where $L_G^{\dagger}$ denotes the pseudo-inverse.*

Proof: Write any $x = \alpha e + y$, where e is the all-ones vector and $\langle y, e\rangle = 0$. Then

$$\begin{aligned}
L_H \preceq (1+\varepsilon)L_G &\iff x^T L_H x \le (1+\varepsilon) x^T L_G x \ \ \forall x \\
&\iff y^T L_H y \le (1+\varepsilon) y^T L_G y \ \ \forall y : \langle y, e\rangle = 0 \\
&\iff z^T L_G^{\dagger/2} L_H L_G^{\dagger/2} z \le (1+\varepsilon) z^T L_G^{\dagger/2} L_G L_G^{\dagger/2} z \qquad \text{using } y = L_G^{\dagger/2} z \text{ or } z = L_G^{1/2} y \\
&\iff z^T L_G^{\dagger/2} L_H L_G^{\dagger/2} z \le (1+\varepsilon) z^T z \ \ \forall z : \langle z, e\rangle = 1 \\
&\iff L_G^{\dagger/2} L_H L_G^{\dagger/2} \preceq (1+\varepsilon) I.
\end{aligned}$$

□

For the lower bound, we have the following analogous result.

Lemma 4 *$L_H \succeq (1-\varepsilon)L_G$ iff $L_G^{\dagger/2} L_H L_G^{\dagger/2} \succeq (1-\varepsilon)I$ iff $\lambda^*_{min}(L_G^{\dagger/2} L_H L_G^{\dagger/2}) \ge (1-\varepsilon)$, where λ^*_{min} is the minimum over all non-zero eigenvalues.*

The proof is similar to above after claiming that we can do something similar to Chernoff bounds using λ^*_{min}.

Proceeding with our discussion of the algorithm, $L_H = \sum_{(i,j)\in E} w(i,j)(e_i - e_j)(e_i - e_j)^T$ so

$$\begin{aligned}
L_G^{\dagger/2} L_H L_G^{\dagger/2} &= \sum_{(i,j)\in E} w(i,j)\Big[L_G^{\dagger/2}(e_i - e_j)(e_i - e_j)^T L_G^{\dagger/2}\Big] \\
&= \sum_{(i,j)\in E} w(i,j) x_{(i,j)} x_{(i,j)}^T \qquad \text{where } x_{(i,j)} = L_G^{\dagger/2}(e_i - e_j).
\end{aligned}$$

Then $(1-\varepsilon)L_G \preceq L_H \preceq (1+\varepsilon)L_G$ is equivalent to showing

$$\lambda_{max}\Big(\sum_{(i,j)\in E} w(i,j) x_{(i,j)} x_{(i,j)}^T\Big) \le (1+\varepsilon), \text{ and}$$

$$\lambda^*_{min}\Big(\sum_{(i,j)\in E} w(i,j) x_{(i,j)} x_{(i,j)}^T\Big) \geq (1-\varepsilon).$$

Note that

$$\begin{aligned}\sum_{(i,j)\in E} x_{(i,j)} x_{(i,j)}^T &= \sum_{(i,j)\in E} L_G^{\dagger/2}(e_i-e_j)(e_i-e_j)^T L_G^{\dagger/2} \\ &= L_G^{\dagger/2}\Big[\sum_{(i,j)\in E}(e_i-e_j)(e_i-e_j)^T\Big]L_G^{\dagger/2} = L_G^{\dagger/2} L_G L_G^{\dagger/2} = I^*,\end{aligned}$$

where I^* is something like the identity (technically, it's the product of $L_G L_G^\dagger$, which when multiplied by any vector x orthogonal to e returns x). And

$$x_{(i,j)}^T x_{(i,j)} = (e_i-e_j)^T L_G^{\dagger/2} L_G^{\dagger/2}(e_i-e_j) = (e_i-e_j)^T L_G^{\dagger}(e_i-e_j) = r_{\text{eff}}(i,j).$$

Now with this setup, we can prove the following theorem.

Theorem 5 *H is a spectral sparsifier of G with probability $\geq 1-\frac{2}{n}$.*

Proof: We want to apply the matrix Chernoff bound. We let $X_k = \frac{n-1}{\ell\cdot r_{\text{eff}}(i,j)} x_{(i,j)} x_{(i,j)}^T$ if we pick edge (i,j) in the kth iteration. Then

$$L_G^{\dagger/2} L_H L_G^{\dagger/2} = \sum_{k=1}^{\ell} X_k \equiv X.$$

Also,

$$\begin{aligned}\mathbf{E}[X] = \sum_{k=1}^{\ell}\mathbf{E}[X_k] &= \sum_{k=1}^{\ell}\sum_{(i,j)\in E}\frac{r_{\text{eff}}(i,j)}{n-1}\cdot\frac{n-1}{\ell\cdot r_{\text{eff}}(i,j)} x_{(i,j)} x_{(i,j)}^T \\ &= \sum_{(i,j)\in E} x_{(i,j)} x_{(i,j)}^T = I^*\end{aligned}$$

We want to show that $0 \preceq X_k \preceq R\cdot I$ for some R and $\ell = 6(n-1)\ln n/\varepsilon^2$. It is clear that $X_k \succeq 0$. $X_k = \frac{n-1}{\ell}\frac{xx^T}{x^Tx}$ for $x = x_{(i,j)}$, so

$$z^T X_k z = \frac{n-1}{\ell}\frac{z^T x x^T z}{x^T x} = \frac{n-1}{\ell}\frac{(x^T z)^2}{x^T x} \leq \frac{n-1}{\ell} z^T z \qquad \text{by Cauchy-Schwarz,}$$

so that $X_k \preceq \frac{n-1}{\ell} I$ means $X_k \preceq R\cdot I$ for $R = \frac{n-1}{\ell} = \frac{\varepsilon^2}{6\ln n}$.

Then using $\mu_{min} = \mu_{max} = 1$ and $\delta = \varepsilon$,

$$\mathbf{Pr}[\lambda_{max}(X) \geq 1+\varepsilon] \leq n\cdot\exp\left(-\frac{\varepsilon^2}{3R}\right) = n\cdot\exp\left(-2\ln n\right) = \frac{1}{n}$$

$$\mathbf{Pr}[\lambda_{min}(X) \leq 1-\varepsilon] \leq n\cdot\exp\left(-\frac{\varepsilon^2}{2R}\right) = n\cdot\exp(-3\ln n) = \frac{1}{n^2} \leq \frac{1}{n}.$$

The theorem statement follows by the union bound. □

In the next lecture, we will see how to use a variation of the multiplicative weight update algorithm to find a spectral sparsifier deterministically.

We did not get to showing how to compute effective resistances, but can be done using a very nice trick involving dimension reduction and the Johnson-Lindenstrauss lemma.