Spectral Graph Theory and Algorithms, Summer 2017	27 June 2017

Lecture 7

Lecturer: David P. Williamson

1 Trevisan's inequality

Recall that we have been looking at the largest eigenvalue of the normalized Laplacian, and that

$$\lambda_n = \max_{x \in \mathbb{R}^n} \frac{x^\top \mathscr{L} x}{x^\top x} = \max_{x \in \mathbb{R}^n} \frac{x^\top D^{-1/2} L_G D^{-1/2} x}{x^\top x} = \max_{y \in \mathbb{R}^n} \frac{y^\top L_G y}{y^\top D y},$$

where we take $y = D^{-1/2}x$. We claimed the following, and proved one direction.

Claim 1 $\lambda_n = 2$ *if and only if* G *has a bipartite component.*

Recall the quantity

$$\beta(G) = \min_{\substack{S \subseteq V \\ S = L \cup R \\ L \cap R = \emptyset}} \frac{2|E(L)| + 2|E(R)| + |\delta(S)|}{\mathrm{vol}(S)},$$

where $E(X)$ denotes the set of edges with both endpoints in X. Alternatively,

$$\beta(G) = \min_{y \in \{-1,0,1\}^n} \frac{\sum_{(i,j)\in E} |y(i) + y(j)|}{\sum_{i \in V} d(i)|y(i)|},$$

where $L = \{i : y(i) = 1\}, R = \{i : y(i) = -1\}$ and $S = L \cup R$.

Since λ_n is the largest eigenvalue of $\mathscr{L}$, $\beta_n = 2 - \lambda_n$ is the smallest eigenvalue of $2I - \mathscr{L} = 2I - (I - \mathscr{A}) = I + \mathscr{A}$. Hence

$$\beta_n = \min_{x \in \mathbb{R}^n} \frac{x^\top (I + \mathscr{A}) x}{x^\top x} = \min_{x \in \mathbb{R}^n} \frac{x^\top D^{-1/2}(D + \mathscr{A}) D^{-1/2} x}{x^\top x} = \min_{y \in \mathbb{R}^n} \frac{y^\top (D + A) y}{y^\top D y};$$

that is,

$$\beta_n = \min_{y \in \mathbb{R}^n} \frac{\sum_{(i,j)\in E} (y(i) + y(j))^2}{\sum_{i \in V} d(i) y(i)^2}.$$

Last time we stated and began the proof of the following result, which is designed to be analogous to Cheeger's inequality.

Theorem 2 (Trevisan 2009)

$$\frac{1}{2}\beta_n \le \beta(G) \le \sqrt{2\beta_n}.$$

[0]This lecture is derived from Lau's 2015 notes, Lecture 4, `https://cs.uwaterloo.ca/~lapchi/cs798/notes/L04.pdf`, from Trevisan Chapter 4, `https://people.eecs.berkeley.edu/~luca/books/expanders.pdf` and Vishnoi, Chapter 8, `http://research.microsoft.com/en-us/um/people/nvishno/site/Lxb-Web.pdf`. Scribes were Victor Reis, Michael Roberts, and Julian Sun.

Proof: We proved the first inequality last time. For the second inequality, pick $y \in \mathbb{R}^n$ satisfying $\beta_n = \frac{y^\top(D+A)y}{y^\top y}$ and assume that $\max_i y^2(i) = 1$ (if this is not true, scale y accordingly). Choose $t \in [0,1]$ uniformly at random, and set $x(i) = 1$ if $x(i) \geq \sqrt{t}$, $x(i) = -1$ if $x(i) \leq -\sqrt{t}$ and $x(i) = 0$ otherwise.

Claim 3 *$\mathbb{E}[|x(i) + x(j)|] \leq |y(i) + y(j)| \cdot (|y(i)| + |y(j)|)$ for all $(i,j) \in E$.*

Proof of claim: Without loss of generality suppose $y(i)^2 \geq y(j)^2$. If $y(i), y(j)$ have the same sign then

$$\begin{aligned}\mathbb{E}[|x(i) + x(j)|] &= 1 \cdot \mathbb{P}[y(j)^2 \leq t \leq y(i)^2] + 2 \cdot \mathbb{P}[t \leq y(j)^2] \\ &= y(i)^2 + y(j)^2 \\ &\leq |y(i) + y(j)| \cdot (|y(i)| + |y(j)|).\end{aligned}$$

Otherwise, $y(i), y(j)$ have different signs, so

$$\begin{aligned}\mathbb{E}[|x(i) + x(j)|] &= 1 \cdot \mathbb{P}[y(j)^2 \leq t \leq y(i)^2] \\ &= y(i)^2 - y(j)^2 \\ &= (y(i) + y(j))(y(i) - y(j)) \leq |y(i) + y(j)| \cdot (|y(i)| + |y(j)|),\end{aligned}$$

as claimed. □

Summing over all $(i,j) \in E$ and using Cauchy-Schwarz gives

$$\begin{aligned}\mathbb{E}\left[\sum_{(i,j)\in E} |x(i) + x(j)|\right] &\leq \sum_{(i,j)\in E} |y(i) + y(j)| \cdot (|y(i)| + |y(j)|) \\ &\leq \sqrt{\sum_{(i,j)\in E} (y(i) + y(j))^2}\sqrt{\sum_{(i,j)\in E} (|y(i)| + |y(j)|)^2} \\ &\leq \sqrt{\beta_n \sum_{i\in V} d(i)y(i)^2}\sqrt{\sum_{(i,j)\in E} 2(y(i)^2 + y(j)^2)} \\ &= \sqrt{2\beta_n}\sum_{i\in V} d(i)y(i)^2 \\ &= \sqrt{2\beta_n}\mathbb{E}\left[\sum_{i\in V} d(i)|x(i)|\right].\end{aligned}$$

Thus we have that

$$\mathbb{E}\left[\sum_{(i,j)\in E} |x(i) + x(j)| - \sqrt{2\beta_n}\sum_{i\in V} d(i)|x(i)|\right] \leq 0,$$

so that there must exist an $t \in (0,1]$ and a corresponding $x \in \{-1,0,1\}^n$ with

$$\beta(G) \leq \frac{\sum_{(i,j)\in E} |x(i) + x(j)|}{\sum_{i\in V} d(i)|x(i)|} \leq \sqrt{2\beta_n},$$

as desired. As with the proof of the Cheeger inequality, we can find such an x easily because there are only n possible different vectors x produced by the algorithm, and these correspond to $t = y(i)^2$ for all $i \in V$. □

2 An application to approximation algorithms for MAX CUT

Recall the MAX CUT problem: Given $G = (V, E)$, find $S \subset V$ that maximizes $\delta(S)$.

Definition 1 (Approximation algorithm) *A (randomized) α-approximation algorithm runs in (randomized) polynomial time and computes a solution with (expected) value within α of the value of an optimal solution.*

Note that there exists an easy randomized algorithm: Flip a coin for each $i \in V$ to decide whether or not $i \in S$. Then

$$\mathbb{E}\left[|\delta(S)|\right] = \sum_{(i,j)\in E} \Pr[(i,j) \in S] = \frac{1}{2}|E| \geq \frac{1}{2}\,\mathrm{OPT},$$

where OPT is the value of an optimal solution to **Max-Cut** on G.

We can now show a .529-approximation algorithm due to Trevisan using a combination of this naive randomized algorithm and Trevisan's inequality. We use the fact that the proof is algorithmic: it finds L, R, and $S = L \cup R$ so that $\beta(S) \leq \sqrt{2\beta_n}$.

The main idea of this algorithm is to trade off between two cases:

- If $\mathrm{OPT} < (1-\epsilon)|E|$, then we get an approximation ratio from the naive random algorithm that is better than 1/2.
- If $\mathrm{OPT} \geq (1-\epsilon)|E|$, then we can use Trevisan's inequality to get a better bound.

For the maximum cut S^*, let $S = V$, $L = S^*$, $R = V - S^*$. Suppose that $\mathrm{OPT} \geq (1-\epsilon)|E|$. Then

$$\begin{aligned}
\beta(G) \leq \beta(S) &= \frac{2|E(S^*)| + 2|E(V - S^*)| + |\delta(V)|}{\mathrm{vol}(V)} = \frac{2(|E| - |\delta(S^*)|)}{2|E|} \\
&\leq \frac{2(|E| - (1-\epsilon)|E|)}{2|E|} \\
&= \epsilon.
\end{aligned}$$

Notice that in this case, then, we can infer that $\beta_n \leq 2\epsilon$.

So if $\beta_n > 2\epsilon$, then $\mathrm{OPT} < (1-\epsilon)|E|$. Then the naive randomized algorithm finds S such that

$$\mathbb{E}\left[\delta(S)\right] = \frac{1}{2}|E| \geq \frac{\mathrm{OPT}}{2(1-\epsilon)}.$$

Thus in this case it is a $\frac{1}{2(1-\epsilon)}$- approximation algorithm.

Now suppose that $\beta_n \leq 2\epsilon$. We can run the algorithm to find a set S and a partition of S into L and R such that $\beta(S)$ is small, namely, at most $\sqrt{2\beta_n} \leq 2\sqrt{\epsilon}$.

Once we have this S, what should we do to find a large cut? In this case, we will attempt to improve our bounds by making some recursive calls. We recurse our Max-Cut algorithm on $V - S$, to find (L', R') that partition $S - V$. Consider the following two possible cuts of G (presented as partitions on V):

- $(L \cup L', R \cup R')$
- $(L \cup R', R \cup L')$

Notice that every edge in $\delta(S)$ either "stays on the same side", going from L to L' or R to R', or else "crosses sides", going from L to R' or R to L'. That means that one of the above cuts must contain at least 1/2 the edges in $\delta(S)$. We choose that cut.

Call the size of the cut our algorithm finds on G, $\mathrm{ALG}(G)$, and the size of the maximum cut in G, $\mathrm{OPT}(G)$. Then:

$$\mathrm{ALG}(G) \geq |\delta(L,R)| + 1/2\delta(S) + \mathrm{ALG}(G-S),$$

and

$$\mathrm{OPT}(G) \leq |E(L)| + |E(R)| + |\delta(L,R)| + |\delta(S)| + \mathrm{OPT}(G-S).$$

Then

$$\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \geq \min\left\{\frac{|\delta(L,R)| + 1/2\delta(S)}{|E(L)| + |E(R)| + |\delta(L,R)| + |\delta(S)|}, \frac{\mathrm{ALG}(G-S)}{\mathrm{OPT}(G-S)}\right\}.$$

Since $\beta_n \leq 2\epsilon$, using Trevisan's inequalities we bound:

$$\begin{aligned}
2\sqrt{\epsilon} &\geq \frac{2|E(L)| + 2|E(R)| + |\delta(S)|}{\mathrm{vol}(S)} \\
&= \frac{2|E(L)| + 2|E(R)| + |\delta(S)|}{2|E(L)| + 2|E(R)| + |\delta(S)| + 2|\delta(L,R)|} \\
&= 1 - \frac{|\delta(L,R)|}{|E(L)| + |E(R)| + 1/2|\delta(S)| + |\delta(L,R)|}.
\end{aligned}$$

Thus

$$\frac{|\delta(L,R)| + 1/2\delta(S)}{|E(L)| + |E(R)| + |\delta(L,R)| + |\delta(S)|} \leq \frac{|\delta(L,R)|}{|E(L)| + |E(R)| + |\delta(L,R)| + 1/2|\delta(S)|} \leq 1-2\sqrt{\epsilon}.$$

So, we can conclude that

$$\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \geq \min\left\{1 - 2\sqrt{\epsilon}, \frac{\mathrm{ALG}(G-S)}{\mathrm{OPT}(G-S)}\right\}.$$

The same must hold true for $G - S$ recursively. But note that for some subgraph of G we consider in some recursive step, it may be possible that $\beta_n \geq 2\epsilon$. Thus we conclude that:

$$\frac{\mathrm{ALG}(G)}{\mathrm{OPT}(G)} \geq \min\left\{1 - 2\sqrt{\epsilon}, \frac{1}{2(1-\epsilon)}\right\}.$$

These two expressions are equal for $\epsilon \approx .0554$, at which point the ratio is about .529. So this is a .529-approximation algorithm.[1]

Better analyses were given in Trevisan 2009, which improved the bound to .531, and in Soto 2015, which improved it to .614.

3 Discussion

Goemans, W (1995) gave a .878-approximation algorithm for MAX CUT by using semidefinite programming (SDP). So why do we care about Trevisan's spectral algorithm?

[1] Lau, in his lecture notes, attributes this analysis to Nick Harvey.

- Computing eigenvectors is a lot easier than solving SDP. (Although, Trevisan's algorithm makes recursive calls that require recomputing new vectors). However, experimentally it seems that Trevisan's algorithm is both faster and better than the SDP-based algorithms, at least if we make a few tweaks. For instance, rather than drawing a single value of t, we can try to draw several and see which gives us a good value of $\beta(S)$ before recursing.
- This method may be more powerful than LP. Kothari, Meka and Raghavedra (STOC 2017) show that subexponentially-sized LPs are required to get better than a 1/2-approximation algorithm for MAX CUT.

These observations raise some research questions:

- The current bound on the algorithm's performance doesn't seem tight - is it?
- Is there a"one-shot" spectral algorithm, one that doesn't require recursive calls? The recursion makes it hard to analyze the algorithm, and forces recomputation of eigenvectors.
- Can we apply this algorithm to other problems with a similar structure (called 2-CSP)? For instance, the MAX DICUT problem (MAX CUT in directed graphs) and the MAX 2SAT problem have this structure. In the MAX 2SAT problem, we are given n boolean variables $x_1, \dots, x_n$, and some number of clauses with at most two variables (e.g. $\bar{x}_1$, $x_2 \vee \bar{x}_3$, etc.) The goal is to find a setting of the variables to true or false so as to maximize the total number of satisfied clauses.

Some progress has been made on this last question.

Definition 2 (Balanced MAX E2SAT) *Balanced MAX E2SAT is a subclass of MAX 2SAT instances such that each clause has exactly two literals in it (i.e. variables or their negations) and for all i, the number of clauses in which x_i appears is exactly equal to the number of clauses in which $\bar{x}_i$ appears.*

Paul, Poloczek, W (2016) use Trevisan's algorithm to obtain a .81-approximation algorithm for Balanced MAX E2SAT, which is better than a .75-approximation algorithm that can be obtained via a naive randomized algorithm.

4 Computing eigenvectors

So far we have not said much about how to actually compute an eigenvector and an eigenvalue. Suppose we want to find the eigenvector corresponding to the largest eigenvalue of a symmetric positive semidefinite matrix A.

Let $\mathbf{x}_1, \dots, \mathbf{x}_n$ be orthonormal eigenvectors, with $\lambda_1 \geq \cdots \geq \lambda_n$ being the corresponding eigenvalues of A. Consider the following algorithm, where $N(0,1)$ is the standard normal distribution of mean 0, standard deviation 1.

If $A \in \mathbb{R}^{n \times n}$ has m nonzero entries, then this algorithm runs in $O(k(m+n))$ time.

The idea is that if we repeatedly multiply $\mathbf{v}$ by A, the largest eigenvalue predominates and the components of eigenvectors that have eigenvalue bounded away from the largest become relatively small. With that in mind, we can show that the vector we have constructed in Power Method has Rayleigh ratio reasonably close to that of the eigenvector of the largest eigenvalue.

Algorithm 1: Power Method

Pick $\mathbf{v}_0$ by drawing $\mathbf{v}_0(i) \sim N(0,1)$ for all i
for $j \leftarrow 1$ **to** k **do**
    $\mathbf{v}_j \leftarrow A\mathbf{v}_{j-1}$
return $\mathbf{v}_k$

Lemma 4 *Let* $\mathbf{v}$ *be such that* $\mathbf{v}(i) \sim N(0,1)$ *for all* i*,* $|\mathbf{v}^T\mathbf{x}_1| \geq \frac{1}{2}$*,* $A \succeq 0$*. Then for* $k > 0$*,* $\epsilon > 0$*,* $\mathbf{y} = A^k\mathbf{v}$*,*

$$\frac{\mathbf{y}^T A\mathbf{y}}{\mathbf{y}^T\mathbf{y}} \geq \lambda_1(1-\epsilon)\frac{1}{1+4\|\mathbf{v}\|^2(1-\epsilon)^{2k}}.$$

If we can show that the lemma is true, then for $k = O(\frac{\log n}{\epsilon})$, $\|\mathbf{v}\|^2 \leq 2n$, we have

$$\frac{\mathbf{y}^T A\mathbf{y}}{\mathbf{y}^T\mathbf{y}} \geq \lambda_1(1-\epsilon)\frac{1}{1+8n(1-\epsilon)^{2k}} \geq \lambda_1(1-\epsilon)\frac{1}{1+\frac{8}{n}} \geq \lambda_1(1-2\epsilon)$$

for n sufficiently large.