Counting and Sampling **Fall 2017**

Lecture 8: Path Technology

Lecturer: Shayan Oveis Gharan *October 20*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications.*

8.1 Dirichlet Form

For two functions f, g define

$$\mathcal{E}(f,g) = \langle (I-K)f, g\rangle$$

The operator (matrix) $I-K$ is also called the Laplacian matrix. Note that this is slightly different from usual Laplacian because $I-K$ is not a symmetric matrix. Next, we study quadratic forms of a Laplacian matrix.

Fact 8.1. $\mathcal{E}(f,g) = \frac{1}{2}\sum_{x,y}(f(x)-f(y))(g(x)-g(y))\pi(x)K(x,y)$.

Proof.

$$\begin{aligned} LHS = \sum_x (I-K)f(x)g(x)\pi(x) &= \sum_x (f(x) - Kf(x))g(x)\pi(x) \\ &= \sum_x f(x)g(x)\pi(x) - \sum_{x,y} K(x,y)f(y)g(x)\pi(x) \end{aligned}$$

On the other hand,

$$\begin{aligned} RHS = \frac{1}{2}\sum_{x,y} f(x)g(x)\pi(x)K(x,y) + \frac{1}{2}\sum_{x,y} f(y)g(y)\pi(y)K(y,x) &- \frac{1}{2}\sum_{x,y} f(y)g(x)K(x,y)\pi(x) \\ &- \frac{1}{2}\sum_{x,y} f(x)g(y)K(y,x)\pi(y) \end{aligned}$$

Relabeling y with x we obtain the equality. □

Therefore,

$$\mathcal{E}(f,f) = \frac{1}{2}\sum_{x,y}(f(x)-f(y))^2\pi(x)K(x,y).$$

$\mathcal{E}(f,f)$ is called the *Dirichlet Form* associated with f.

Definition 8.2 (Variance)**.** *For a function f let,*

$$\mathrm{Var}(f) \quad = \quad \|f - \mathbb{E}_\pi[f]\|^2 .$$

where $\mathbb{E}_\pi[f]$ is the average of f with respect to π.

The following is immediate:

Fact 8.3. *For any function f,*

$$\mathrm{Var}(f) = \frac{1}{2}\sum_{x,y}(f(x)-f(y))^2\pi(x)\pi(y).$$

Proof.

$$\begin{aligned}
\mathrm{Var}(f) &= \sum_x \pi(x)f(x)^2 - \mathbb{E}_\pi[f]^2 \\
&= \frac{1}{2}\sum_x \pi(x)f(x)^2 + \frac{1}{2}\sum_y \pi(y)f(y)^2 - \sum_{x,y} f(x)f(y)\pi(x)\pi(y) \\
&= \frac{1}{2}\sum_{x,y}(f(x)-f(y))^2\pi(x)\pi(y)
\end{aligned}$$

□

Now, we are ready for the following important definition:

Definition 8.4 (Poincaré Constant)**.** *For a (non-necessarily reversible) kernel K the Poincare constant is defined as*

$$\min_f \frac{\mathcal{E}(f,f)}{\mathrm{Var}(f)},$$

where the minimum is taken over all functions f with nonzero variance.

Lemma 8.5. *If K is reversible, then the Poincaré constant is equal to $1-\lambda_2$.*

Proof. We leave the proof of this as an exercise. It simply follows from the the variational characterization of eigenvalues. □

Now observe that when we make a chain lazy, we are working with the $(I+K)/2$ as a Markov kernel. Since all eigenvalues of K are at least -1, $(I+K)/2$ is a PSD operator, i.e., all eigenvalues are nonnegative. So, λ^* is equal to λ_2. Therefore, it follows from Theorem 7.6 that for a lazy (reversible) chain with Poincaré constant α, and any starting state x,

$$\tau_x(\epsilon) \le O\left(\frac{\log\left(\frac{1}{\epsilon\cdot\pi(x)}\right)}{\alpha}\right)$$

There is a very different proof of this fact that holds for any not necessarily reversible chain. The proof goes by showing that for lazy Markov chain and any function f,

$$\mathrm{Var}(Kf) \le \mathrm{Var}(f) - \mathcal{E}(f,f).$$

From this one can deduce that $\mathrm{Var}(K^t f) \le (1-\alpha)^t \cdot \mathrm{Var}(f)$ where α is the Poincaré constant. Therefore, for $t = \log(\frac{1}{\pi(x)})/\alpha$ we get that the chain mixes.

8.2 Path Technology

Next we discuss a new method to bound the Poincaré constant. Note that by Theorem 7.6 this gives a direct upper bound on the mixing time.

Think of a multicommodity flow problem where we want to send $\pi(x)\pi(y)$ unit of flow between each pair of states x, y. Note that these are supposed to be disjoint commodities. This means that if a 1 unit $x \to y$ flow goes on edge (u, v) from u to v and 1 unit of $x' \to y'$ flow goes on the same edge from v to u they don not cancel out each other. The capacity of each edge $e = (u, v)$ is $Q(e) = \pi(u)K(u, v) = \pi(v)K(v, u)$. Recall that $Q(e)$ represents the flow of the probability mass along edge e at stationarity. Suppose we choose a path $P_{x,y}$ between each pair of vertices x, y (note that more generally this may be a distribution of paths). For an edge e let flow of e be defined as follows:

$$f(e) = \sum_{x,y:e\in P_{x,y}} \pi(x)\pi(y).$$

The congestion of e is defined as $\frac{f(e)}{Q(e)}$. In the following lemma we show that the inverse of the Poincaré is at most $\max_{x,y} |P_{x,y}| \cdot \max_e \frac{f(e)}{Q(e)}$.

Lemma 8.6. *For any reversible Markov chain, suppose for every pair of states $x, y \in \Omega$ we choose a path $P_{x,y}$.*

$$\frac{1}{\alpha} \leq \max_e \frac{f(e)}{Q(e)} \cdot \max_{x,y} |P_{x,y}|.$$

where α is the Poincaré constant.

Proof. Consider an arbitrary function f. Note that for every path $P_{x,y}$ we can choose an orientation of the edges of this say e^+ is the head and e^- is the tail such that $f(x) - f(y) = \sum_{e\in P_{x,y}} f(e^+) - f(e^-)$. We can write

$$\begin{aligned}
\mathrm{Var}(f) &= \frac{1}{2}\sum_{x,y}(f(x) - f(y))^2\pi(x)\pi(y) \\
&= \frac{1}{2}\sum_{x,y}\left(\sum_{e\in P_{x,y}} f(e^+) - f(e^-)\right)^2 \pi(x)\pi(y) \\
&\leq \frac{1}{2}\sum_{x,y}|P_{x,y}| \sum_{e\in P_{x,y}} (f(e^+) - f(e^-))^2\pi(x)\pi(y) \\
&= \frac{1}{2}\sum_{e=(x',y')} (f(x') - f(y'))^2 \frac{Q(e)}{Q(e)} \sum_{x,y:e\in P_{x,y}} |P_{x,y}|\pi(x)\pi(y) \leq max_{x,y}|P_{x,y}| \cdot \max_e \frac{f(e)}{Q(e)} \cdot \mathcal{E}(f, f).
\end{aligned}$$

where the inequality follows by Cauchy-Schwarz inequality. □

Note that we typically let $P_{x,y}$ be a shortest path from x to y. So, $P_{x,y}$ is no more than the diameter. Since in all Markov chains that we constant diameter is only a polynomial in the size of the input, usually, the most important parameter to bound the mixing time is the maximum congestion.

Let us use the above machinery to bound the mixing time of a simple lazy random walk on a path. Consider the a simple path of length n. The stationarity distribution is almost uniform. Observe that there is a unique path between each pair of vertices. So, the edge $(n/2, n/2+1)$ has the maximum congestion of about

$$\frac{(n/2)(n/2)\frac{1}{n}\frac{1}{n}}{\frac{1}{n}\frac{1}{4}} \approx n.$$

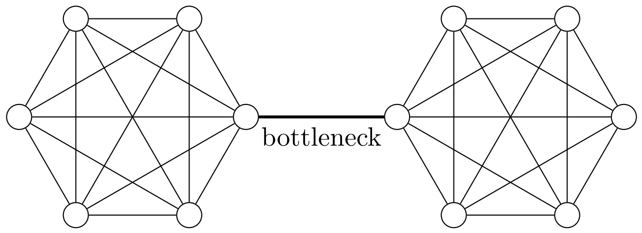
Since $\mathrm{diam}(G) = n$, we get

$$\frac{1}{\alpha} \le \frac{1}{n^2}.$$

It follows that the chain mixes in $O(n^2 \log(n))$. Note that this bound is $O(\log n)$ off from the bound we proved using strong stationarity time. The reason is that here we are just upper bound the second eigenvalue of K and we use a very crude bound on all other eigenvalues (we are upper bound each λ_i by λ_2). So, in many applications of the Path technology this $O(\log(n))$ loss in inherent, but usually it does not chain the mixing time significantly, because it is logarithmic in the size of the state space.

Dumbell Graph Note that if we have a Markov chain with a diameter that is polynomial in the size of the state space, then we should expect a very slow mixing. This is essentially what happens in the path example. So, one can ask if we have a Markov chain with a logarithmic size diameter, can it still have a mixing time polynomial in the size of the instance? The answer is yes, and the famous Dumbell graph is perhaps the worst example. In this graph the single edge which is connecting the two cliques will be a bottleneck because all of the flow between the two cliques must go over this edge.

Note that although we didn't prove, but the path technology gives a necessary and sufficient condition for bounding the Poincaré constant. Namely, if we can show that for any routing of the multicommodity flow there is an edge with large congestion, it would imply that the chain mixes very slowly.



8.3 All or Nothing Theorem

In this section we use the path technology to prove the all or nothing theorem in counting.

Theorem 8.7. *For any self-reducible problem, if there exists a polynomial time counting algorithm that gives* $1 + \mathrm{poly}(n)$ *multiplicative factor approximation, then there is an FPRAS.*

First, let us proof a weaker version of this theorem. Suppose we can get a constant factor α approximation for a counting problem, say coloring. So, for a given graph G if it has C possible coloring the algorithm returns a number in $p(G)$ where

$$\frac{C}{\alpha} \le p(G) \le \alpha C.$$

Now, we feed this algorithm k disjoint copies of G, say $G^{\oplus k}$. Then, the number of solutions to this instance is of course C^k. So, the algorithm must still return a number in $p(G^{\oplus k}$ where

$$\frac{C^k}{\alpha} \le p(G^{\oplus K}) \le \alpha C^k.$$

It follows that

$$\frac{C}{\alpha^{1/k}} \le p(G^{\oplus K})^{1/k} \le \alpha^{1/k} C.$$

So, we can to $1+\epsilon$ letting $k = \log(\alpha)/\epsilon$. Next, we will do the general case. The above simple idea does not work if the approximation factor depends on n, because once we below up the instance the approximation factor deteriorates.

Proof. Say we work with the matching counting problem as a template. Recall that we consider an ordering of all edges of G, say $e_1, \dots, e_m$. We consider a tree where every node at level i corresponds to a (valid) configuration for the first i edges, whether they are forced to be in or out. For example for $i = 1$, we have two graph G_1 where we delete the endpoints of e_1 and G_2 where we delete the edge e. For any node (graph) z of this tree, let N_z denote the number of matchings of the graph corresponding to z or equivalently, the number of leaves of the subtree rooted at z, T_z. We also assume that we have access to an oracle that for any such z returns an estimate $\tilde{N}_z$ where

$$\frac{N_z}{\alpha} \le \tilde{N}_z \le \alpha N_z$$

for some $\alpha = \text{poly}(n)$. For simplicity we assume that the counting oracle is deterministic. Our goal is to build a uniform sampler. We can then use the sampler to construct an FPRAS.

For each edge (y, z) in the tree where y is the parent of z, we let $w_{y,z} = \tilde{N}_z$. Now, we run a random walk on this weighted tree starting from the root. That is at each vertex we choose an adjacent edge proportional to its weight. Note that the leafs of the tree correspond to matchings all of the have weight 1 (we don't need to run the counting oracle on the leaves). Note that we may also move upward the tree.

Here is the high-level intuition for this random walk: The downward moves correspond to the previous idea that we had for designing a sampling algorithm. Unfortunately, if the error in counting is large the multiplicative errors accumulate as we go down the tree, so we may end up selecting a matching with probability c^m larger than the stationary distribution. Roughly speaking the upward edges in the Markov chain is to correct these incorrect estimates. The following three facts imply that the Markov chain can be used to generate a random matching

Fact 8.8. *The stationary distribution π is uniform over all leaves.*

This is simply because the weight of the single edge incident to every leaf is 1. Recall that in the stationary distribution of a random walk in an undirected weighted graph, the probability of every vertex is proportional to its (weighted) degree. This fact implies that if the chain mixes, and at stationarity we sample a state which happens to be a leaf, that leaf gives us a uniformly random matching. Next, we study the probability of sampling a leaf in the stationary distribution.

Fact 8.9. *The sum of the weight of leaves is at least $\frac{O(1)}{\alpha m}$, i.e., a sample $X \sim \pi$, with probability $\frac{O(1)}{\alpha m}$ is a leaf.*

Let N be the correct solution that we are trying to find, i.e., the tree has N leaves. It follows that

$$\sum_{x \text{ is a leaf}} \pi(x) = \frac{1}{D} = \frac{N}{D},$$

where D is the sum of the degree of all nodes of the tree. Now observe that the sum of the edge weights of each level of the tree is at most $\alpha|\mathcal{M}(G)|$. This is because every estimate returned by the approximate counter is within α factor of the number of leaves in the corresponding subtree. It follows that

$$D \le 2\alpha m N. \tag{8.1}$$

Therefore, sum of the probability of the leaves at stationarity is at least

$$\frac{N}{D} \ge \frac{1}{2\alpha m}.$$

The above fact shows that if we generate $O(\alpha m)$ samples of the chain at mixing, then with high probability we obtain a uniformly random leave, i.e., a uniformly random matching.

So, to prove the theorem, it remains to bound the mixing time of the chain. We show that the chain mixes in $O(\alpha^2 m^2 \log(\alpha m))$ steps. This implies that we can obtain a uniformly random matching by running the chain for $O(\alpha^3 m^3 \log(\alpha m))$ steps which is a polynomial in m, n.

To bound the mixing time we use the path technology. Note that there is a unique path between each pair of vertices of the tree. So, there is no choice in constructing the paths. Let us just calculate the congestion of every edge. Note that the longest path has length $O(m)$. So, it is enough to show that the maximum congestion is at most $O(\alpha^2 m)$. Fix an edge $e = (z', z)$ where z' is the parent of z. First observe that

$$Q(e) = \pi(z)\frac{\tilde{N}_z}{D(z)} = \frac{D(y)}{D} \cdot \frac{\tilde{N}_z}{D(y)} = \frac{\tilde{N}_z}{D}.$$

It remains to upper bound $f(e)$. Let T_z be the tree rooted at z. Then,

$$f(e) = \sum_{x \in T_z, y \notin T_z} \pi(x)\pi(y) \leq \sum_{x \in T_z} \pi(x) = \pi(T_z).$$

Using a calculation similar to what we did before,

$$\pi(T_z) \leq \frac{1}{D} 2\alpha m N_z.$$

This is because the sum of the edge weights of each level of T_z is at most αN_z. It follows that

$$\frac{f(e)}{Q(e)} \leq \frac{2\alpha m N_z}{\tilde{N}_z} \leq 2\alpha^2 m.$$

Therefore, the inverse of the Poincaré constant is at most $O(\alpha^2 m^2)$. And, the chain mixes in $O(\alpha^2 m^2 \log \pi(root))$. But

$$\pi(root) \geq \frac{N/\alpha}{D} \geq \frac{1}{2\alpha^2 m},$$

where we used (8.1). □