Chapter 4

Introduction to Spectral Graph Theory

Spectral graph theory is the study of a graph through the properties of the eigenvalues and eigenvectors of its associated Laplacian matrix. In the following, we use $G = (V, E)$ to represent an undirected n-vertex graph with no self-loops, and write $V = \{1, \ldots, n\}$, with the degree of vertex i denoted d_i. For undirected graphs our convention will be that if there is an edge then both $(i, j) \in E$ and $(j, i) \in E$. Thus $\sum_{(i,j)\in E} 1 = 2|E|$. If we wish to sum over edges only once, we will write $\{i, j\} \in E$ for the unordered pair. Thus $\sum_{\{i,j\}\in E} 1 = |E|$.

4.1 Matrices associated to a graph

Given an undirected graph G, the most natural matrix associated to it is its *adjacency matrix*:

Definition 4.1 (Adjacency matrix)**.** The adjacency matrix $A \in \{0, 1\}^{n\times n}$ is defined as

$$A_{ij} = \begin{cases} 1 & \text{if } \{i, j\} \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Note that A is always a symmetric matrix with exactly d_i ones in the i-th row and the i-th column. While A is a natural representation of G when we think of a matrix as a table of numbers used to store information, it is less natural if we think of a matrix as an operator, a linear transformation which acts on vectors. The most natural operator associated with a graph is the *diffusion operator*, which spreads a quantity supported on any vertex equally onto its neighbors. To introduce the diffusion operator, first consider the *degree matrix*:

Definition 4.2 (Degree matrix)**.** The degree matrix $D \in \mathbb{R}^{n\times n}$ is defined as the diagonal matrix with diagonal entries $(d_1, \ldots, d_n)$.

Definition 4.3 (Normalized Adjacency Matrix)**.** The normalized adjacency matrix is defined as

$$\overline{A} = AD^{-1}.$$

Note that this is not necessarily a symmetric matrix. But it is a column-stochastic matrix: each column has non-negative entries that sum to 1. This means that if $p \in \mathbb{R}^n_+$ is a probability vector defined over the vertices, then $\overline{A}p$ is another probability vector, obtained by "randomly walking" along an edge, starting from a vertex chosen at random according to p. We will explore the connection between $\overline{A}$ and random walks on G in much more detail in the next lecture.

Finally we introduce the *Laplacian* matrix, which will provide us with a very useful quadratic form associated to G:

Definition 4.4 (Laplacian and normalized Laplacian Matrix)**.** The Laplacian matrix is defined as

$$L = D - A.$$

The normalized Laplacian is defined as

$$\overline{L} = D^{-1/2} L D^{-1/2} = \mathbb{I} - D^{-1/2} A D^{-1/2}.$$

Note that L and $\overline{L}$ are always symmetric. They are best thought of as quadratic forms: for any $x \in \mathbb{R}^n$,

$$x^T L x = \sum_i d_i x_i^2 - \sum_{(i,j)\in E} x_i x_j = \sum_{\{i,j\}\in E} (x_i - x_j)^2.$$

For the normalized Laplacian, we have the following claim.

Claim 4.5. *$\forall x \in \mathbb{R}^n$, we have*

$$x^T \overline{L} x = \sum_{\{i,j\}\in E} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2. \tag{4.1}$$

If G is d-regular, then this simplifies to

$$x^T \overline{L} x = \frac{1}{d} \sum_{\{i,j\}\in E} (x_i - x_j)^2.$$

Proof.

$$\begin{aligned}
x^T \overline{L} x &= x^T x - x^T D^{-1/2} A D^{-1/2} x \\
&= \sum_i x_i^2 - \sum_{i,j} \frac{x_i}{\sqrt{d_i}} A_{ij} \frac{x_j}{\sqrt{d_j}} \\
&= \sum_i d_i \left(\frac{x_i}{\sqrt{d_i}} \right)^2 - \sum_{(i,j)\in E} \frac{x_i}{\sqrt{d_i}} \cdot \frac{x_j}{\sqrt{d_i}} \\
&= \sum_{\{i,j\}\in E} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2.
\end{aligned}$$

□

Claim 4.5 provides the following interpretation of the Laplacian: if we think of the vector x as assigning a weight, or "potential" $x_i \in \mathbb{R}$ to every vertex $v \in V$, then the Laplacian measures the average variation of the potential over all edges. The expression $x^T \overline{L} x$ will be small when the potential x is close to constant across all edges (when appropriately weighted by the corresponding degrees), and large when it varies a lot, for instance when potentials associated with endpoints of an edge have a different sign.

We will return to this interpretation soon. Let's first see some examples. It will be convenient to always order the eigenvalues of A in deceasing order, $\mu_1 \geq \cdots \geq \mu_n$, and those of $\overline{L}$ in increasing order, $\lambda_1 \leq \cdots \leq \lambda_n$. So what do the eigenvalues of $\overline{A}$ or $\overline{L}$ have to say?

Example 4.6. Consider the graph shown in Figure. 4.1.

Figure 4.1: A single edge

The adjacency matrix is

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Note that in writing down A we have some liberty in ordering the rows and columns. But this does not change the spectrum as simultaneous reordering of the rows and the columns corresponds to conjugation by a permutation, which is orthogonal and thus preserves the spectral decomposition. We can also compute

$$D = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad L = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \overline{L}.$$

The spectrum of $\overline{L}$ is given by $\lambda_1 = 0$, $\lambda_2 = 2$. The corresponding eigenvectors are

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad \text{and} \quad \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Example 4.7. Consider the graph shown in Figure. 4.2.

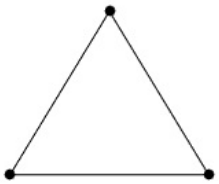


Figure 4.2: The triangle graph

The adjacency matrix is given by

$$A = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

We can also compute

$$D = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix} \quad \text{and} \quad L = \begin{pmatrix} 1 & -1/2 & -1/2 \\ -1/2 & 1 & -1/2 \\ -1/2 & -1/2 & 1 \end{pmatrix}.$$

The eigenvalues of $\overline{L}$ are $0, 3/2, 3/2$ with corresponding eigenvectors

$$\frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad \frac{1}{\sqrt{6}} \begin{pmatrix} 2 \\ -1 \\ -1 \end{pmatrix}, \quad \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix},$$

where since the second eigenvalue $3/2$ is degenerate we have freedom in choosing a basis for the associated 2-dimensional subspace.

Example 4.8. As a last example, consider the path of length two, pictured in Figure. 4.3.

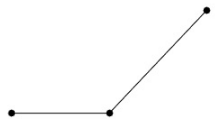


Figure 4.3: The path of length 2

The adjacency matrix is given by

$$A = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

We can also compute

$$D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad L = \begin{pmatrix} 1 & -1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1 & -1/\sqrt{2} \\ 0 & -1/\sqrt{2} & 1 \end{pmatrix}.$$

The eigenvalues of $\overline{L}$ are $0, 1, 2$ with corresponding eigenvectors

$$\frac{1}{2} \begin{pmatrix} 1 \\ \sqrt{2} \\ 1 \end{pmatrix}, \quad \frac{1}{\sqrt{2}} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}, \quad \frac{1}{2} \begin{pmatrix} -1 \\ \sqrt{2} \\ -1 \end{pmatrix}.$$

We've seen three examples — do you notice any pattern? 0 seems to always be the smallest eigenvalue. Moreover, in two cases the associated eigenvector has all its coefficients equal. In the case of the path, the middle coefficient is larger — this seems to reflect the degree distribution in some way. Anything else? The largest eigenvalue is not always the same. Sometimes there is a degenerate eigenvalue.

Exercise 1. Show that the largest eigenvalue λ_n of the normalized Laplacian of a connected graph G is such that $\lambda_n = 2$ if and only if G is bipartite.

We will see that much more can be read about combinatorial properties of G from $\overline{L}$ in a systematic way. The main connection is provided by the Courant-Fisher theorem:

Theorem 4.9 (Variational Characterization of Eigenvalues)**.** *Let $M \in \mathbb{R}^{n\times n}$ be a symmetric matrix with eigenvalues $\mu_1 \geq \cdots \geq \mu_n$, and let the corresponding eigenvectors be $v_1, \cdots, v_n$. Then*

$$\mu_1 = \sup_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{x^T M x}{x^T x}, \quad \mu_2 = \sup_{\substack{x\in\mathbb{R}^n \\ x\perp v_1}} \frac{x^T M x}{x^T x}, \quad \dots \quad \mu_n = \sup_{x\in\mathbb{R}^n \atop x\perp v_1,\dots,v_{n-1}} \frac{x^T M x}{x^T x} = \inf_{\substack{x\in\mathbb{R}^n \\ x\neq 0}} \frac{x^T M x}{x^T x}.$$

Proof. By the spectral theorem, we can write

$$M = \sum_{i=1}^{n} \mu_i v_i v_i^T, \tag{4.2}$$

where $\{v_1, \dots, v_n\}$ are an orthonormal basis of $\mathbb{R}^n$ formed of eigenvectors of M. For $1 \leq k \leq n$, we have

$$\mu_k \leq \sup_{\substack{x\in\mathbb{R}^n \\ x\perp v_1,\dots,v_{k-1}}} \frac{x^T M x}{x^T x} \tag{4.3}$$

because by taking $x = v_k$ and using (4.2) together with $v_i^T v_k = 0$ for $i \neq k$ we immediately get

$$\frac{x^T M x}{x^T x} = \mu_k.$$

To show the reverse inequality, observe that any x such that $x^T x = 1$ and $x \perp v_1, \dots, v_{k-1}$ can be decomposed as $x = \sum_{j=k}^{n} \alpha_j v_j$ with $\sum_j \alpha_j^2 = 1$. Now

$$x^T M x = \sum_{i,j=k}^{n} \sum_{l=1}^{n} \mu_l \alpha_i \alpha_j v_i^T v_l v_l^T v_j = \sum_{l=k}^{n} \mu_l \alpha_l^2 \leq \mu_k$$

since the eigenvalues are ordered in decreasing order. Thus

$$\mu_k \geq \sup_{\substack{x\in\mathbb{R}^n \\ x\perp v_1,\dots,v_{k-1}}} \frac{x^T M x}{x^T x},$$

which together with (4.3) concludes the proof. □

4.2 Eigenvalues of the Laplacian

Using the variational characterization of eigenvalues given in Theorem 4.9, we can connect the quadratic form associated with the normalized Laplacian in Claim 4.5 to the eigenvalues of $\overline{L}$.

Claim 4.10. *For any graph G with normalized Laplacian $\overline{L}$, $0 \leq \overline{L} \leq 2\mathbb{I}$. Moreover, if λ_1 is the smallest eigenvalue of $\overline{L}$ then $\lambda_1 = 0$ with multiplicity equal to the number of connected components of G.*

Proof. From (4.1) we see that $x^T\overline{L}x \geq 0$ for any x, and using $(a-b)^2 \leq 2(a^2+b^2)$ we also have $x^T\overline{L}x \leq 2x^Tx$. Using the variational characterization

$$\lambda_1 = \inf_{x \neq 0} \frac{x^T\overline{L}x}{x^Tx}, \qquad \lambda_n = \sup_{x \neq 0} \frac{x^T\overline{L}x}{x^Tx},$$

where λ_n is the largest eigenvalue, we see that $0 \leq \overline{L} \leq 2\mathbb{I}$. To see that $\lambda_1 = 0$ always with multiplicity at least 1 it suffices to consider the vector

$$v_1 = \begin{pmatrix} \sqrt{d_1} \\ \vdots \\ \sqrt{d_n} \end{pmatrix},$$

for which $v_1^T\overline{L}v_1 = 0$.

Now suppose G has exactly L connected components. By choosing a vector equal to $\sqrt{d_i}$ for all i that belong to a given connected component and 0 elsewhere we can construct as many orthogonal vectors v such that $v^T\overline{L}v = 0$ as there are connected components. Thus the multiplicity of the eigenvalue 0 is at least as large as the number of connected components.

To show the converse, note that from (4.1) we see that up to normalization any v such that $v^T\overline{L}v = 0$ must be such that $v_i/\sqrt{d_i}$ is constant across each connected component. Thus the dimension of the subspace of all v such that $v^T\overline{L}v = 0$ is effectively at most the number of connected components, and there can be at most k linearly independent such vectors: the multiplicity of the eigenvalue 0 is at most the number of connected components. □

An immediate corollary worth stating explicitly is as follows:

Claim 4.11. *For any graph G, the second smallest eigenvalue $\lambda_2(\overline{L}) > 0$ if and only if G is connected.*

These claims show that the small eigenvalues of $\overline{L}$ tell us whether the graph is connected or not. We will make this statement more quantitative by showing that, not only is the question of connectedness related to the question of λ_2 being equal to 0, but in fact the magnitude of λ_2 can be used to quantify, in a precise way, how "well-connected" the graph is. Let us look at a natural measure of connectedness of a graph, its *conductance*. Given a set of vertices $\emptyset \subsetneq S \subsetneq V$, the boundary of S is defined as

$$\partial S = \big\{\{i,j\} \in E : i \in S, j \notin S\big\}.$$

The conductance of S is

$$\phi(S) = \frac{|\partial S|}{\min\left(d(S), d(V \backslash S)\right)},$$

where $d(S) := \sum_{i \in S} d_i$ is a natural measure of volume: the total number of edges incident on vertices in S. If G is d-regular, then this simplifies to

$$\phi(S) = \frac{|\partial S|}{d \cdot \min(|S|, |V \backslash S|)}.$$

Definition 4.12 (Conductance)**.** The conductance of a graph G is defined as

$$\phi(G) = \min_{S : S \neq \emptyset, S \neq V} \phi(S).$$

If G is d-regular, this simplifies to

$$\Phi(G) = \min_{S,\, 1 \leq |S| \leq n/2} \frac{|\partial S|}{d \cdot |S|},$$

the fraction of edges incident on S that have one endpoint outside of S.

The conductance is a measure of how well connected G is. Here are some examples demonstrating this point.

Example 4.13.

- Clearly G is disconnected if and only if there exists a set $S \neq \emptyset$, $S \neq V$ such that $|\partial S| = 0$, i.e. if and only if $\phi(G) = 0$.
- If G is a clique, then

$$\phi(G) = \min_{1 \leq k \leq n/2} \frac{k(n-k)}{(n-1)k} = \frac{n}{2(n-1)} \approx \frac{1}{2}.$$

- If G is a cycle, then

$$\phi(G) = \min_{1 \leq k \leq n/2} \frac{2}{2k} = \frac{2}{n}.$$

Exercise 2. Compute the conductance of the hypercube $G = (V, E)$ where $V = \{0,1\}^n$ and $E = \{\{u, v\} \in V : d_H(u, v) = 1\}$, where d_H is the Hamming distance.

The following theorem is a fundamental result relating conductance and the second smallest eigenvalue of the normalized Laplacian.

Theorem 4.14 (Cheeger's inequality)**.** *Let G be an undirected graph with normalized Laplacian $\overline{L} = \mathbb{I} - D^{-1/2} A D^{-1/2}$. Let $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ be the eigenvalues of $\overline{L}$. Then*

$$\frac{\lambda_2}{2} \leq \phi(G) \leq \sqrt{2\lambda_2}.$$

Remark 4.15.

- Both sides of the inequality are interesting. The left-hand side says that if there is a good cut, that is a cut of small conductance, then there is an eigenvector orthogonal to the smallest eigenvector with small eigenvalue. This is called the "easy" side of Cheeger.
- The right-hand side says that if λ_2 is small, then there must exist a poorly connected set. This is called the "hard" side of Cheeger.
- We will give "algorithmic" proofs of both inequalities: for the left-hand side, given a set S of low conductance we will show how to construct a vector $v \perp v_1$ that achieves a low value in (4.1). For the right-hand side, given a vector $v_2 \perp v_1$ achieving a low value in (4.1) we will construct a set S of low conductance.
- The next exercise shows that both sides of the inequality are tight.

Exercise 3. Show that the left-hand side of Cheeger's inequality is tight by computing the eigenvalues and eigenvectors of the hypercube (hint: Fourier basis). Show that the right-hand side is also tight by considering the example of the n-cycle.

Proof of Cheeger's inequality. We first prove the "*easy side*". Let S be a set of vertices such that $\phi(S) = \phi(G)$. We claim

$$\lambda_2 = \min_{\substack{x \in \mathbb{R}^n \\ x \perp v_1 = \left(\sqrt{d_1}, \cdots, \sqrt{d_n}\right)}} \frac{x^T \overline{L} x}{x^T x} \leq 2\phi(G).$$

To see this, define

$$x = (\underbrace{\sqrt{d_i}, \ldots}_{\text{vertices in } S}, \underbrace{-\sigma\sqrt{d_j}, \ldots}_{\text{vertices in } \overline{S}}),$$

where $\sigma = \frac{d(S)}{d(V \setminus S)}$ is defined so that

$$x^T v_1 = \sum_{i \in S} d_i - \sigma \sum_{i \in \overline{S}} d_i = 0.$$

We then have

$$x^T x = \sum_{i \in S} d_i + \sigma^2 \sum_{i \in V \setminus S} d_i = d(S) + \frac{d(S)^2}{d(V \setminus S)^2} \cdot d(V \setminus S) = \frac{d(S) d(V)}{d(V \setminus S)},$$

and

$$\begin{aligned} x^T \overline{L} x &= \sum_{\{i,j\} \in E} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2 \\ &= \sum_{\{i,j\} \in \partial S} (1 + \sigma)^2 = \sum_{\{i,j\} \in \partial S} \left(\frac{d(V \setminus S) + d(S)}{d(V \setminus S)} \right)^2 \\ &= |\partial S| \frac{d(V)^2}{d(V \setminus S)^2}. \end{aligned}$$

This finally implies

$$\frac{x^T \overline{L} x}{x^T x} = \frac{|\partial S| d(V)}{d(S) d(V \backslash S)} \leq \frac{2|\partial S|}{\min(d(S), d(V \backslash S))} = 2\phi(G),$$

where the inequality can be seen by considering the cases $d(S) \leq d(V \backslash S)$ and $d(S) > d(V \backslash S)$ separately.

Now let's turn to the "*hard side*" of the inequality. Let $y \in \mathbb{R}^n$ be such that

$$\frac{y^T \overline{L} y}{y^T y} \leq \lambda_2 \tag{4.4}$$

and $y \perp v_1 = (\sqrt{d_1}, \ldots, \sqrt{d_n})^T$. Our main idea is going to be to think of the coordinates of y as providing an ordering of the vertices, with each coordinate telling us how likely (small, negative y_i) or unlikely (large, positive y_i) the vertex is to be in a set with small conductance: this intuition comes from the form of the vector we found in the proof of the easy case, which provides such an ordering.

Rather than use the inequality (4.4) as starting point, it will be more convenient to work with the analogue formulation for the Laplacian L, so we start with a few manipulations. Let $z = D^{-1/2} y - \sigma \mathbf{1}$ for some σ to be determined soon and $\mathbf{1} = (1, \ldots, 1)^T$. Since $\mathbf{1}^T L \mathbf{1} = 0$, we see that $z^T L z = y^T \overline{L} y$. Moreover, $D\mathbf{1} = v_1$, thus

$$z^T D z = y^T y - 2\sigma \underbrace{v_1^T \cdot y}_{=0} + \sigma^2 d(V) \geq y^T y,$$

and using (4.4) we get that for any σ,

$$\frac{z^T L z}{z^T D z} \leq \lambda_2.$$

We make the following conventions, without loss of generality:

- Order the coordinates of z so that $z_1 \leq \ldots \leq z_n$.
- Choose σ such that $z_{i_0} = 0$, where i_0 is such that

$$\sum_{i < i_0} d_i < \frac{d(V)}{2} \qquad \sum_{i \leq i_0} d_i \geq \frac{d(V)}{2}. \tag{4.5}$$

- Scale z so that $z_1^2 + z_n^2 = 1$.

Let $t \in [z_1, z_n]$ be chosen according to the distribution with density $2|t|$ (the scaling on z assumed above ensures that this is a properly normalized probability measure). Observe

that for any $a < b$,

$$\begin{aligned}\mathbf{Pr}(t \in [a,b]) &= \int_a^b 2|t|dt \\ &= \begin{cases} b^2 + a^2 & \text{if } a < 0 < b \\ b^2 - a^2 & \text{if } b > a > 0 \\ a^2 - b^2 & \text{otherwise} \end{cases} \\ &\leq |b-a|\left(|a| + |b|\right),\end{aligned}$$

an inequality that is easily verified in all three cases. For any t, let $S_t = \{i : z_i \leq t\}$. Then

$$\mathop{\mathbf{E}}_t d(S_t) = \sum_i \mathbf{Pr}(i \in S_t) d_i = \sum_i \mathbf{Pr}(z_i \leq t) d_i.$$

Our choice of the index i_0 in (4.5) ensures that, if $t < 0$ then $\min(d(S_t), d(V\backslash S_t)) = d(S_t)$, while if $t \geq 0$ then $\min(d(S_t), d(V\backslash S_t)) = d(V\backslash S_t)$. Thus

$$\begin{aligned}\mathop{\mathbf{E}}_t \min\left(d(S_t), d(V\backslash S_t)\right) &= \sum_{i<i_0} \mathbf{Pr}(z_i \leq t \wedge t < 0) d_i + \sum_{i \geq i_0} \mathbf{Pr}(z_i > t \wedge t \geq 0) d_i \\ &= \sum_{i<i_0} z_i^2 d_i + \sum_{i \geq i_0} z_i^2 d_i \qquad (4.6) \\ &= z^T D z. \qquad (4.7)\end{aligned}$$

Next we compute

$$\begin{aligned}\mathop{\mathbf{E}}_t |\partial S_t| &= \sum_{\{i,j\} \in E} \mathbf{Pr}\left(z_i \leq t \leq z_j\right) \\ &\leq \sum_{\{i,j\} \in E} |z_j - z_i|(|z_i| + |z_j|) \\ &\leq \underbrace{\sqrt{\sum_{\{i,j\} \in E} (z_i - z_j)^2}}_{\sqrt{z^T L z}} \quad \underbrace{\sqrt{\sum_{\{i,j\} \in E} (|z_i| + |z_j|)^2}}_{\leq \sqrt{2\sum_{\{i,j\} \in E}(|z_i|^2 + |z_j|^2)} = \sqrt{2 z^T D z}} \\ &\leq \sqrt{2\lambda_2} z^T D z, \qquad (4.8)\end{aligned}$$

where the second inequality is Cauchy-Schwarz. Combining (4.6) and (4.8),

$$\mathop{\mathbf{E}}_t |\partial S_t| \leq \sqrt{2\lambda_2} \mathop{\mathbf{E}}_t \min\left(d(S_t), d(v\backslash S_t)\right),$$

which we can rewrite as

$$\mathop{\mathbf{E}}_t \left[\sqrt{2\lambda_2} \min\left(d(S_t), d(v\backslash S_t)\right) - |\partial S_t|\right] \geq 0.$$

From there we deduce that there exists a choice of t such that

$$\Phi(S_t) \leq \sqrt{2\lambda_2},$$

which immediately gives us that $\Phi(G) \leq \sqrt{2\lambda_2}$, as desired. □

We note that the proof given above is *algorithmic*, in that it describes an efficient algorithm that, given a graph, will produce a set with conductance at most $\sqrt{2\lambda_2}$: simply compute an eigenvector associated with the second smallest eigenvalue (using, e.g., the power method), and output the set S_t which has smallest conductance among the n possibilities: this can be checked efficiently.

4.3 Sparsity

The *sparsity* is another natural measure of "disconnectedness" of a graph, which turns out to be very closely related to the conductance.

Definition 4.16. For a d-regular graph G and $\emptyset \subsetneq S \subsetneq V$, the sparsity of S is defined as

$$\sigma(S) = \frac{\mathbf{E}_{(i,j)\in E}\,|1_S(i) - 1_S(j)|}{\mathbf{E}_{(i,j)\in V^2}\,|1_S(i) - 1_S(j)|} = \frac{\frac{1}{dn}|\partial S|}{\frac{2}{n^2}|S|\cdot|V\backslash S|} = \frac{|V||\partial S|}{2d|S||V\backslash S|},$$

and so

$$\frac{\Phi(S)}{2} \leq \sigma(S) \leq \Phi(S).$$

Note that the definition of the sparsity lets us give a different, almost immediate proof of the "easy" side of Cheeger's inequality:

$$\begin{aligned}
\sigma(G) &= \min_S \sigma(S) \\
&= \min_{x\in\{0,1\}^n, x\neq\mathbf{0},\mathbf{1}} \frac{\frac{1}{dn}\sum_{(i,j)\in E}|x_i - x_j|^2}{\frac{2}{n^2}\sum_{(i,j)\in V^2}|x_i - x_j|^2} \\
&\geq \min_{x\in\mathbb{R}^n, x\neq\mathbf{0}, x\perp\mathbf{1}} \frac{2\sum_{\{i,j\}\in E}(x_i - x_j)^2}{\frac{2d}{n}\sum_{(i,j)\in V^2}(x_i - x_j)^2} \\
&= \min_{x\in\mathbb{R}^n, x\neq\mathbf{0}, x\perp\mathbf{1}} \frac{x^T\overline{L}x}{\frac{1}{n}\sum_{(i,j)\in V^2}(x_i - x_j)^2} \\
&= \min_{x\in\mathbb{R}^n, x\neq\mathbf{0}, x\perp\mathbf{1}} \frac{x^T\overline{L}x}{\frac{1}{n}\sum_{i\in V} 2nx_i^2 - 2\sum_{i,j} x_i x_j} \\
&= \min_{x\in\mathbb{R}^n, x\neq\mathbf{0}, x\perp\mathbf{1}} \frac{x^T\overline{L}x}{2x^Tx} = \frac{\lambda_2}{2},
\end{aligned}$$

where here the inequality follows since we are taking the minimum over a larger set (the constraint $x \perp \mathbf{1}$ is without loss of generality since the whole expression is invariant under translation of the vector x by an additive constant multiple of $\mathbf{1}$).

Using the above inequality $\sigma(G) \leq 2\Phi(G)$, we have re-proven the left-hand side of Cheeger's inequality, simply by observing that the second eigenvalue of $\overline{L}$ could be seen as a natural *relaxation* of the sparsity, itself very closely related to the conductance.

The second eigenvalue gives us a good approximation to the conductance in case λ_2 is not too small, say it is a constant. If however λ_2 goes to 0 with n, say $\lambda_2 \sim 1/n$, then the approximation can be very bad, it can be a multiplicative factor $\lambda_2^{-1/2} \sim \sqrt{n}$ off.

Here are two other relaxations that have been considered, and do much better. The first one is due to Leighton and Rao, and can be defined as

$$\mathrm{LR}(G) = \min_{\substack{w \in \mathbb{R}^{n \times n} \\ w_{ij} \geq 0, w_{ii} = 0 \\ w_{ij} \leq w_{ik} + w_{kj}}} \frac{\sum_{(i,j) \in E} w_{ij}}{\frac{d}{n} \sum_{(i,j) \in V^2} w_{ij}}. \tag{4.9}$$

This can be interpreted as a minimization over all *semi-metrics*: distance measures $d(i,j) = w_{ij}$ on the graph that are always non-negative and satisfy the triangle inequality. An example such metric is $(i,j) \to w_{ij} = |x_i - x_j|$ (for any fixed vector x), but there are others. The advantage of allowing all semi-metrics is that $\mathrm{LR}(G)$ can be computed using a linear program. Moreover, Leighton and Rao showed that

$$O(\log n)\,\mathrm{LR}(G) \geq \sigma(G) \geq \mathrm{LR}(G),$$

thus we get a much tighter approximation to the sparsity than the one given by λ_2 in cases when the sparsity is small. An even tighter relaxation was introduced by Arora, Rao and Vazirani, who considered

$$\mathrm{ARV}(G) = \min_{\substack{w \in \mathbb{R}^{n \times n} \\ w_{ij} \geq 0, w_{ii} = 0 \\ w_{ij}^2 \leq w_{ik}^2 + w_{kj}^2}} \frac{\sum_{(i,j) \in E} w_{ij}}{\frac{d}{n} \sum_{(i,j) \in V^2} w_{ij}}. \tag{4.10}$$

This is the same as before, except now we require d^2, instead of d, to be a semi-metric. This is a stronger requirement, and it is still a relaxation because $d_{ij} = |x_i - x_j|$ satisfies both conditions. The optimization problem defining $\mathrm{ARV}(G)$ can be solved efficiently using semidefinite programming, and Arora, Rao and Vazirani showed that

$$O(\sqrt{\log n})\,\mathrm{ARV}(G) \geq \sigma(G) \geq \mathrm{ARV}(G).$$

It is open whether one can find a better approximation to $\sigma(G)$ in polynomial time. The best hardness results are Unique Games hardness for constant-factor approximations, and anything in-between is open!

4.4 Random walks on graphs

4.4.1 A motivating example

We start with a motivating application. The problem k-SAT is, given m clauses over n Boolean variables such that each clause is a disjunction of literals (a literal is a variable or

its negation), is it possible to find an assignment to the variables that satisfies all clauses? For $k \geq 3$ the problem is NP-hard, but for $k = 2$ there are efficient algorithms.

Here is a simple candidate. Start with a random assignment to the variables. At each step, choose a clause that is not satisfied. Pick one of the two variables it acts on at random, and flip it. Repeat.

Is this going to work? And if so, how long will take? Here is the key idea. Suppose there exists a satisfying assignment, and fix it. Now consider the distance between the current assignment, maintained by the algorithm, and this satisfying assignment. This distance is an integer between 0 and n, and at each step it either increases or decreases by 1. If a clause is violated, at least one of the two variables involved must have a different value in the current assignment as in the satisfying assignment. With probability $1/2$ we flip this variable, so that at each step with probability at least $1/2$ we decrease the distance by 1. How many steps will it take to find the satisfying assignment? The answer is $O(n^2)$, and we'll see how to show this very easily once we've covered some of the basics of the analysis of random walks on arbitrary graphs. (Note the algorithm we just described is not the best, and it is possible to solve 2SAT in deterministic linear time...)

4.4.2 The random walk matrix

Let G be a undirected weighted graph with adjacency matrix A. Put $d_i = \sum_{j:\{i,j\}\in E} w_{ij}$ (we will always assume all weights to be non-negative). The natural random walk on G is to step from $i \to j$ with probability $\frac{w_{ij}}{d_i}$. Let $p^{(0)} \in \mathbb{R}^n_+$ be a distribution over vertices. One step of the random walk is that

$$p^{(0)} \to p^{(1)}, \quad \text{where} \quad p^{(1)}_j = \sum_{i:\{i,j\}\in E} p^{(0)}_i \frac{w_{ij}}{d_i},$$

which in matrix form can be written as $p^{(1)} = AD^{-1}p^{(0)}$. This version of the random walk has one major drawback, which is that it does not always converge: consider for instance a graph with a single edge, or more generally a bipartite graph; the walk started at a vertex on the left will continue hopping back and forth between left and right without ever converging. To overcome this issue it is customary to consider instead the *lazy* random walk: with probability $1/2$, do not move, and with probability $1/2$, do as before. The update rule is then

$$p^{(t)} = \left(\frac{\mathbb{I}}{2} + \frac{1}{2}AD^{-1}\right) p^{(t-1)},$$

which in matrix form is $p^{(t)} = Wp^{(t-1)}$ where W is the random walk matrix:

Definition 4.17. Let $G = (V, E)$ be an n-vertex weighted, undirected graph with weights $w_{ij} \geq 0$. Define the *lazy random walk* on G: if $p \in \mathbb{R}^n_+$ is a distribution on the vertices $V = \{1, ..., n\}$, one step of the random walk brings p to Wp where

$$W = \frac{1}{2}\mathbb{I} + \frac{1}{2}AD^{-1}.$$

Note that W is not symmetric, so it is not diagonalizable. But observe that

$$W = D^{1/2}\left(\mathbb{I} - \frac{1}{2}\left(\mathbb{I} - D^{-1/2}AD^{-1/2}\right)\right)D^{-1/2} = D^{1/2}\Big(\mathbb{I} - \frac{\overline{L}}{2}\Big)D^{-1/2},$$

$\overline{L}$ is the normalized Laplacian associated with G. Thus if v is an eigenvector for $\overline{L}$ with eigenvalue λ, then $w = D^{1/2}v$ is a right eigenvector for W with eigenvalue $(1-\lambda/2)$; thus W has n eigenvalues $w_i = 1-\lambda_i/2$ that are directly related to those of the normalized Laplacian. This will let us transfer our understanding of the λ_i to derive convergence properties of the random walk.

4.4.3 Mixing Time

Let $p^{(0)}$ be any distribution on $V = \{1, \ldots, n\}$, and for every $t \geq 1$ define $p^{(t)} = Wp^{(t-1)}$ inductively as the distribution on vertices after t steps of the lazy random walk have been performed. The main question is: does $\lim_{t\to\infty} p^{(t)}$ exist, and if so, how fast is convergence?

Definition 4.18. A distribution π is a *stationary distribution* if $W\pi = \pi$.

By definition, any stationary distribution π is such that π is an eigenvector of W with eigenvalue 1, thus $D^{1/2}\pi$ is a right eigenvector of $\overline{L}$ with eigenvalue 0. But we know that if G is connected $\overline{L}$ always has a single non-degenerate eigenvalue equal to 0, and moreover the associated eigenvector $v_1 \propto (\sqrt{d_1}, \ldots, \sqrt{d_n})$ can be taken to have all its components non-negative. Thus there exists a unique stationary distribution $\pi \propto D^{1/2}(\sqrt{d_1}, \ldots, \sqrt{d_n}) \propto (d_1, \ldots d_n)$; normalizing the vector π so that it indeed corresponds to a distribution we obtain that $\pi = \frac{1}{d(V)}(d_1, \ldots d_n)$.

Definition 4.19. Given $\varepsilon > 0$, the *mixing time* is defined as

$$\tau_\varepsilon \;=\; \min\big\{t : \|W^t p - \pi\|_1 \leq \varepsilon \,\forall p \in \mathbb{R}^n_+ \text{ s.t.} \sum_i p_i = 1\big\},$$

where π is the stationary distribution, and $\|\cdot\|_1$ is the statistical distance. By convention we also let $\tau = \tau_{1/4}$.

Theorem 4.20. *For any connected, weighed, undirected graph G the lazy random walk mixes in time $\tau_\varepsilon = O(\frac{\log(n/\varepsilon)}{\lambda_2})$, where λ_2 is the second-smallest eigenvalue of $\overline{L}$.*

Proof. Take p any distribution on V. Then, $p = \sum_{i=1}^N \alpha_i D^{1/2} v_i$, where v_i are the eigenvectors of $\overline{L}$ and α_i some arbitrary coefficients. Here, $v_1 = d(V)^{1/2}D^{-1/2}\pi$, so $\alpha_1 = (D^{-1/2}p)^T v_1 = d(V)^{-1/2}$ since p is a normalized distribution. Then,

$$\begin{aligned} W^t p &= \sum \alpha_i W^t D^{1/2} v_i \\ &= \sum_i \alpha_i w_i^t D^{1/2} v_i \\ &= \pi + \sum_{i\geq 2} \alpha_i w_i^t D^{1/2} v_i. \end{aligned}$$

Therefore,

$$\begin{aligned}\|W^t p - \pi\|_1 &= \Big\| \sum_{i\geq 2} D^{1/2} w_i^t \alpha_i v_i \Big\|_1 \\ &\leq \sqrt{n} \Big\| \sum_{i\geq 2} \alpha_i w_i^t D^{1/2} v_i \Big\|_2 \\ &\leq \sqrt{n}\sqrt{d_{max}} \Big(\sum_{i\geq 2} \alpha_i^2 w_i^{2t} \Big)^{1/2} \\ &\leq \sqrt{n}\sqrt{n}\, \omega_2^t,\end{aligned}$$

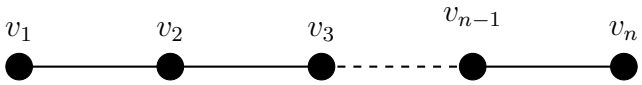
where for the last step we bounded the maximum degree by n and used that $\sum_{i\geq 2} \alpha_i^2 \leq p^T D^{-1} p \leq \sum_i p_i^2 \leq 1$ since p is a distribution. The bound is $\leq \varepsilon$ for $t = \frac{\log(n/\varepsilon)}{\log(w_2)} = \frac{\log(n/\varepsilon)}{\log(1-\lambda_2/2)} = O(\frac{\log(n/\varepsilon)}{\lambda_2})$, as desired. □

Example 4.21. The following simple test cases show that, aside from the $\log n$ factor we lost by switching from the Euclidean norm to the statistical distance, the bound provided by Theorem 4.20 can be very accurate.

- For the n-vertex clique K_n, we saw that the conductance $\phi(K_n) \approx 1/2$ and thus by Cheeger λ_2 is at least a constant. If we put all the probability at time 0 on a single vertex, it will take $\sim \log 1/\varepsilon$ steps before the lazy random walk spreads at least $1-\varepsilon$ probability to the other vertices, so $\tau_\varepsilon = \Omega(\log 1/\varepsilon)$.

- For the n-vertex path P_n, we saw $\lambda_2 \sim 1/n^2$, and you can check that $\tau_\varepsilon \geq n^2 \log(\frac{1}{\varepsilon})$: starting on the leftmost vertex, it takes $\sim n^2$ steps before we hit the rightmost vertex, and the $\log(1/\varepsilon)$ term is overhead before the walk becomes sufficiently close to uniform. Note this proves the bound on the convergence time for the randomized algorithm for 2-SAT we saw earlier.

- For the dumbbell graph $K_{n/3}$ — $P_{n/3}$ — $K_{n/3}$ (two $(n/3)$-cliques linked by a path of length $n/3$), the conductance is at most $\sim 1/n^2$ (cut in the middle), so by Cheeger $\lambda_2 = O(1/n^2)$. Consider a random walk starting on the left-most vertex. Then, one step leads to a uniform distribution on the left $K_{n/3}$ clique. From there the probability to enter the bridge is $\sim 1/n^2$, and moreover the probability of making it through the bridge without falling back into the left clique is about $\sim 1/n$. Using this intuition you can show that the mixing time is of order n^3, so that in fact $\lambda_2 = O(1/n^3)$.



How bad can it get? From the definition we see that the conductance $\phi(G)$ is always at least $1/n^2$, so by Cheeger's inequality as long as G is connected we have $\lambda_2 = \Omega(n^{-4})$. In fact we can prove something slightly better.

Claim 4.22. *Let G be a connected, unweighted graph, λ_2 the second smallest eigenvalue of the normalized Laplacian and r the diameter of G. Then $\lambda_2 \geq \frac{2}{r(n-1)^2}$.*

Proof. For any two vertices u, v let $\overline{E}_{u,v}$ be the normalized Laplacian associated to the graph having a single edge $u \to v$, and $\overline{P}_{u,v}$ the normalized Laplacian for a path of length at most r from u to v. Then $\overline{E}_{u,v} \leq r\overline{P}_{u,v}$, as can be seen from the associated quadratic forms: $(x_u - x_v)^2 \leq r((x_u - x_{u_1})^2 + \cdots + (x_{u_{r-1}} - x_v)^2)$. Since any pair of vertices are connected by a path of length at most r in G, $\overline{L}_{K_n} \leq r\binom{n}{2}\overline{L}_G$, where K_n is the clique on n vertices. Since the normalized Laplacian for K_n has second smallest eigenvalue $\frac{n}{n-1}$, we get the claimed bound on λ_2.

□

4.5 Volume estimation

We make a little detour to explore a neat application of random walks — to the problem of *volume estimation*. First, let's discuss a little bit the general framework for sampling and counting problems.

4.5.1 Approximate counting and sampling

A *counting problem* is specified by a function $f : \Sigma^* \to \mathbb{N}$, where Σ is a finite alphabet. For instance, f could be the function that takes as input a (properly encoded) 3-SAT formula φ and returns the number of satisfying assignments to φ. Given an input $x \in \Sigma^*$, the goal is to return $f(x)$.

A *sampling problem* is a function $f : \Sigma^* \to (S, \pi)$ which returns a pair formed by a finite set S and a distribution π on S. Given an input $x \in \Sigma^*$, the goal is to produce an element $s \in S$ distributed according to π.

Counting and sampling problems tend to be very hard. For instance, even though 2-SAT can be solved in polynomial time (as we saw in the previous lecture), the problem of counting the number of solutions to a 2-SAT formula is $\#P$-hard, where $\#P$ is the analogue of NP for counting problems. The same holds for counting the number of cycles in an undirected graph. On the other hand, counting the number of matchings is known to be in P, via a simple computation.

So we settle for approximation schemes. For counting problems this is called a *fully polynomial randomized approximation scheme* (FPRAS). For sampling it is called a *fully polynomial almost uniform sampler* (FPAUS). More formally,

Definition 4.23. Given a counting problem f, a FPRAS for f is a randomized algorithm which given as input x, ε, and δ returns a value y such that with probability at least $1 - \delta$ (over the algorithm's private random coins), $(1 - \varepsilon)f(x) \leq y \leq (1 + \varepsilon)f(x)$ in time that is polynomial in $|x|$, $1/\varepsilon$, and $\log(1/\delta)$.

Given a sampling problem f, a FPAUS for f is a randomized algorithm which given as input x and δ runs in time polynomial in x and $\log 1/\delta$ and returns an element $s \in S$

that is distributed according to a distribution $p = p(x, \delta)$ such that $\|p - \pi\|_1 \leq \delta$, where $f(x) = (S, \pi)$.

A famous theorem by Jerrum, Valiant and Vazirani shows that approximate counting and approximate sampling are essentially equivalent:

Theorem 4.24. *For "nicely behaved" counting problems (the technical term is "downward self-reducible"), the existence of an FPRAS is equivalent to the existence of an FPAUS.*

Proof sketch. For concreteness we prove the theorem for the problem of counting the number of satisfying assignments to any formula φ.

FPAUS $\Longrightarrow$ FPRAS. Take a polynomial number of satisfying assignments for φ sampled by the FPAUS. This lets us estimate p_0 and p_1, the probability that φ is satisfiable conditioned on $x_1 = 0$ and $x_1 = 1$ respectively. Assume $p_0 \geq 1/2$, the other case being symmetric. Make a recursive call to approximate the number of satisfying assignments to $\varphi_{|x_1=0}$. Let $\hat{N}_0$ be the estimate returned, and output $\hat{N}_0/p_0$.

It is clear that this is correct on expectation. Moreover, using that p_0 is not too small a Chernoff bound shows that the estimate obtained from the FPAUS samples will be very accurate with good probability. It is then not hard to prove by induction that provided a polynomial number of samples are taken at each of the n recursive calls (where n is the number of variables), the overall estimate can be made sufficiently accurate, with good probability.

FPRAS $\Longrightarrow$ FPAUS. First we run the FPRAS to obtain good estimates for the number of satisfying assignments N to φ and N_0 to $\varphi_{|x_1=0}$. We can assume the estimates returned are such that $\hat{N}_0/\hat{N} \geq 1/2$, as otherwise we exchange the roles of 0 and 1. Next we flip a coin with bias $\hat{N}_0/\hat{N}$. If it comes up heads we set $x_1 = 0$ and recurse; if it comes up tails we set $x_1 = 1$ and recurse. Provided the estimates $\hat{N}, \hat{N}_0, \ldots$ are accurate enough the distribution on assignments produced by this procedure will be close in statistical distance to the uniform distribution on satisfying assignments. □

4.5.2 Volume estimation

Let $K \subseteq \mathbb{R}^n$ be convex. Our goal is to estimate the volume $\mathrm{Vol}_n(K)$: we would like to devise an efficient procedure that, given K, returns two real numbers α, β that are such that $\alpha \leq \mathrm{Vol}_n(K) \leq \beta$ and β/α is as small as possible. Before diving into this, however, a basic question — how is K specified?

In order to abstract out the details and provide an algorithm that works in a general context we will assume that the only access we are given to K is through one of the following type of "oracle":

- *Membership oracle* (resp. *weak membership oracle*): given a query $x \in \mathbb{R}^n$, the oracle answers whether $x \in K$ (resp. $x \in K$ or $d(x, K) \geq \varepsilon$; the oracle is allowed to fail whenever neither condition is satisfied).

- *Separation oracle* (resp. *weak separation oracle*): given a query x, the oracle returns the same answer as the membership oracle, but in case $x \notin K$ (resp. $d(x, K) \geq \varepsilon$) it also returns an $y \in \mathbb{R}^n$ such that $y^T z > y^T x \ \forall z \in K$ (resp. $y^T z > y^T x - \varepsilon/2 \ \forall z \in K$).

It is always possible to derive a weak separation oracle from a weak membership oracle, and in this lecture we won't worry about the difference — in fact all we'll need is a weak membership oracle.

But is this enough? What if we query x, and we learn $x \notin K$, with $y = (1, 0, \ldots, 0)$? Then we know we should increase x_1. Say we double every coordinate, to $x' = 2x$. But suppose we get the same answer, again and again. We never know how far K is! It seems like a boundedness assumption is necessary, so we'll assume the following: there exists (known) values $r, R > 0$ such that $B_\infty(0, r) \subseteq K \subseteq B_\infty(0, R)$ with $R/r < 2^{\mathrm{poly}(n)}$, where $B_\infty(0, r)$ denotes the ball of radius r for the ℓ_∞ norm (a square box with sides of length $2r$). In fact, by scaling we may as well assume $r = 1$, and for simplicity we'll also assume $R = n^2$. It is not so obvious at first this is without loss of generality, but it is — with some further re-scaling and shifting of things around it is not too hard to reduce to this case.

(The idea is to slowly grow a simplex inside K. At each step we can perform a change of basis so that $\mathrm{Conv}(e_1, \ldots, e_n) \subseteq K$. Then for $i = 1, \ldots, n$ we check if there is a point $x \in K$ such that $|x_i| \geq 1 + 1/n^2$. If so we include it, rescale, and end up with a simplex of volume at least $1 + 1/n^2$ times the previous one, and this guarantees we won't have to go through too many steps. If there is no such point, we stop, as we've achieved the desired ratio. Finally we need to check that we can find such point, if it exists, in polynomial time; this is the case as if $x \in K$ then $(0, \ldots, 0, x_i, 0, \ldots, 0) \in K$ as well (using convexity and the assumption that K contains the simplex), so it suffices to call the membership oracle n times.)

Now that we have a proper setup in place, can we estimate $\mathrm{Vol}(K)$? Say we only want a multiplicative approximation. An idea would be to use the separation oracle to get some kind of a rough approximation of the boundary of K using hyperplanes, then do some kind of triangulation, and estimate the volume by counting tetrahedrons (or, in dimension n, simplices). In fact there is a very strong no-go theorem for deterministic algorithms:

Theorem 4.25. *For any deterministic polynomial-time algorithm such that on input a convex body $K \subseteq \mathbb{R}^n$ (specified via a separation oracle) returns $\alpha(K)$, $\beta(K)$ such that $\alpha(K) \leq \mathrm{Vol}(K) \leq \beta(K)$, it must be that there exists a constant $c > 0$ and a sequence of convex bodies $\{K_n \subseteq \mathbb{R}^n\}_{n \geq 1}$ such that for all $n \geq 1$,*

$$\frac{\beta(K_n)}{\alpha(K_n)} \geq \Big(\frac{cn}{\log n}\Big)^n.$$

This is very bad: even an approximation within an *exponential* factor is ruled out! Note however that a key to the above result is that the only access to K is given by a separation oracle — if we have more knowledge about K then a polynomial-time algorithm might be feasible (though we don't know any).

Proof idea. The idea for the proof is to design an oracle that answers the queries made by any deterministic algorithm in a way that is consistent with the final convex body being one

of two possible bodies, K or K°, whose volume ratio is exponentially large; if we manage to do this then the algorithm cannot provide an estimate that will be accurate for both K and K°.

The oracle is very simple: upon any query $x \in \mathbb{R}^n$ it is very generous and says that $x/\|x\| \in K$, $-x/\|x\| \in K$, and moreover K is included in the "slab" $\{y : -\|x\| \leq \langle y, x\rangle \leq \|x\|\}$. Note that these answers are all consistent with K being the Euclidean unit ball.

Now, if points $x_1, \ldots, x_m$ have been queried, define K to be the convex hull of $\pm x_i/\|x_i\|$, $\pm e_j$ where e_j are the unit basis vectors. Define $K^\circ = \{y : \langle y, z\rangle \leq 1 \, \forall z \in K$. Then you can check that the oracle's answers are all consistent with K and K°. But their volumes are very different, and one can show that $\mathrm{Vol}(K^\circ)/\mathrm{Vol}(K)$ is roughly of order $(n/\log(m/n))^n$; as long as m is not exponential in n this is exponentially large. □

If we allow randomized algorithms the situation is much better:

Theorem 4.26 (Dyer, Frieze, Kannan 1991)**.** *There exists a fully polynomial randomized approximation scheme for approximating Vol(K).*

A fully polynomial randomized approximation scheme (FPRAS) means that $\forall \varepsilon, \delta > 0$ the algorithm returns a $(1 \pm \varepsilon)$-multiplicative approximation to the volume with probability at least $1 - \delta$, and runs in time $\mathrm{poly}(n, 1/\varepsilon, \log 1/\delta)$. Volume estimation is one of these relatively rare problems for which we have strong indication that randomized algorithms can be exponentially more efficient than deterministic ones (primality testing used to be another such problem before the AKS algorithm was discovered!).

The algorithm of Dyer, Frieze and Kannan had a running time that scaled like $\sim n^{23}$. Since then a lot of work has been done on the problem, and the current record is $\sim n^5$. In principle it's possible this could be lowered even more, to say $\sim n^2$; there are no good lower bounds for this problem.

Proof sketch. The main idea is to use random sampling. For instance, suppose we'd place K inside a large, fine grid, as in Figure 4.4.

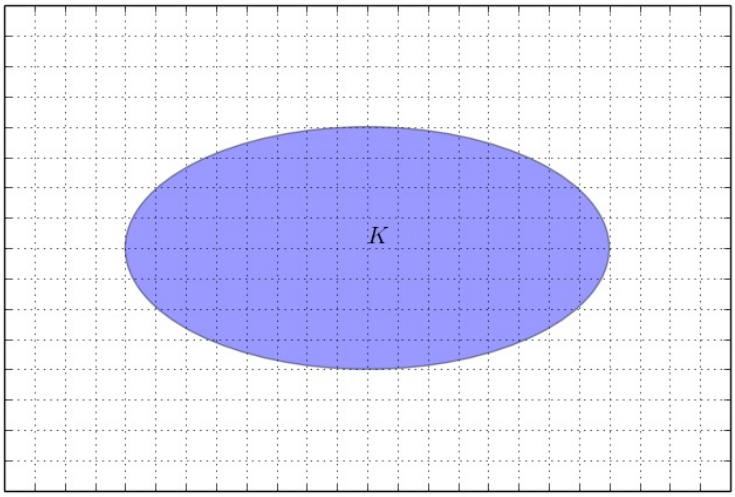


Figure 4.4: The region K is the feasible region.

We could then run a random walk on the grid until it mixes to uniform. This will take time roughly $n(R/\delta)^2$, where δ is the grid spacing; given we assumed $r = 1$ something like $\delta = 1/n^2$ would be reasonable.

Exercise 4. Show that the mixing time of the lazy random walk on $\{1, \ldots, N\}^n$, the n-dimensional grid with sides of length N, is $O(n^2N^2)$.

At the end of the walk we can call the membership oracle to check if we are in K. Since $\text{Prob}(x \in K) \sim \frac{\text{Vol}_n(K)}{\text{Vol}_n(\text{grid})}$, by repeating the walk sufficiently many times we'd get a good estimate. While this works fine in two dimensions (you can estimate $\pi = \text{Area(unit disk)}$ in this way!), in higher dimensions it fails dramatically, as all we know about $\text{Vol}(K)$ is that it is at least 2^n (since it contains the unit ball for ℓ_∞), but the grid could have volume as large as $(2R)^n = (2n^2)^n$, so even assuming perfectly uniform mixing the probability that we actually obtain a point in K is tiny, it's exponentially small.

This is still roughly how we'll proceed, but we're going to have to be more careful. There are three important steps. Here is a sketch:

- Step 1: Subdivision.
  Set $K_0 = B_\infty(0,1) \cap K = B_\infty(0,1), K_1 = B_\infty(0, 2^{1/n}) \cap K, \ldots, K_{2n\log n} = B_\infty(0, n^2) \cap K = K$. Then

$$\text{Vol}(K) = \text{Vol}(K_{2n\log n}) = \frac{\text{Vol}(K_{2n\log n})}{\text{Vol}(K_{2n\log n-1})} \cdot \frac{\text{Vol}(K_{2n\log n-1})}{\text{Vol}(K_{2n\log n-2})} \cdots \frac{\text{Vol}(K_2)}{\text{Vol}(K_1)} \cdot 2^n,$$

since $\text{Vol}(K_1) = 2^n$. So, we have reduced our problem to the following: given $K \subseteq L$ both convex such that $\text{Vol}(K) \geq \frac{1}{2}\text{Vol}(L)$, estimate $\text{Vol}(K)/\text{Vol}(L)$. This eliminates the "tiny ratio" issue we had initially, but now we have another problem: the enclosing set L is no longer a nice grid, but it is an arbitrary convex set itself. Are we making any progress?

- Step 2: A random walk.
  Our strategy will be as follows. Run a random walk on a grid that contains L, such that the stationary distribution of the random walk satisfies the two conditions that $\mathbf{Pr}(x \in L)$ is not too small, and the stationary distribution is close to uniform, conditioned on lying in L. If we can do this we're done: we repeatedly sample from the stationary distribution sufficiently many times that we obtain many samples in L, and we check the fraction of these samples that are also in K: $\frac{\text{Vol}(K)}{\text{Vol}(L)} \sim \frac{\mathbf{Pr}(x \in K)}{\mathbf{Pr}(x \in L)} = \mathbf{Pr}(x \in K | x \in L)$.

  So the challenge is to figure out how to define this random walk around L. Here is a natural attempt. Start at an arbitrary point $x^{(0)} \in L$, say the origin. Set $x^{(1)}$ to be a random neighbor of $x^{(0)}$ on the grid, subject to $x^{(1)} \in L$ (we have $2n$ neighbors to consider, and for each we can call the membership oracle for L). Repeat sufficiently many times. This is the right idea — note that we really want to stay as close to L as possible, because if we allow ourselves to go outside too much we'll get this "tiny ratio" issue once more — but the boundary causes a lot of problems:

  (a) Some points are never reached. L could be very pointy, in which case there could be a grid point that lies in L, but none of its neighbors does. And this cannot be solved just by making the grid finer; it is really an issue with the kinds of angles that are permitted in L.

  (b) The degree of the graph underlying our walk is not constant (it tends to be smaller close to the boundary), so the stationary distribution will not be uniform.

  (c) Some grid cubes have much bigger intersection with L than others.

  It turns out we can fix all of these issues by doing a bit of "smoothing out" on L. Let δ be the width of the grid, and consider $L' = (1 + \delta\sqrt{n})L$, where $\delta\sqrt{n}$ is the diameter of a cube. Assuming $\delta \leq n^{-2}$, this doesn't blow up the volume by much, so in terms of volume ratio we're fine. Moreover, you can check easily that:

  – Any grid point inside L has all of its neighbors in L',
  – All $p \in L$ belong to a grid cube $\subseteq L'$.

  These two points get rid of issue (a) above: all points in L are now reached by the walk. Moreover, we can easily get rid of the degree issue by adding self loops. This guarantees that the stationary distribution will be uniform on grid points in L, clearing (b). Remains (c), the issue of uneven intersection between grid cubes and L. For this we do the following:

  – Do a random walk on $\delta\mathbb{Z}^n \cap L'$ as described before.

– Arrive at a random grid point p. Choose random vector $q \in B_\infty(0,1)$ and output the point $p + \delta q$ if it is in L. Otherwise, restart the walk.

As a result the stationary distribution is uniform on L: if we have a cube C that partially intersects L its points are sampled with probability precisely $\sim \text{Vol}(L \cap C)$. We also need to make sure that there are not too many restarts — what if $\text{Vol}(L \cap C)/\text{Vol}(C)$ is again tiny? Now here we can show this is ok if it happens, because the point is we're trying to estimate the whole volume of L, not just the volume for that intersection. If it's tiny, we never see it, but that's fine.

- Step 3: Mixing Time.
So far the random walk would work even if K is not convex, in the sense that as long as the walk mixes it will let us estimate the ratio $\text{Vol}(K \cap L)/\text{Vol}(L)$. But now we need to understand the conductance of our graph: by Cheeger's inequality and the analysis of mixing time we saw in the previous lecture, as long as the conductance is at least 1/poly, we are good to go.

  Since the graph is regular, given a set of vertices S such that $|S| \leq n/2$, we have $\phi(S) = \frac{|\partial S|}{d|S|}$. If V is the volume of a cube, then $d|S| \sim \frac{\text{Vol}(S)}{V}$, where $\text{Vol}(S)$ is the sum of volumes of cubes at vertices in S, and $|\partial S| \sim \frac{\text{Area}(\partial S)}{A}$, where A is the surface area ($(n-1)$-volume) of a single cube. Since $V/A = 2\delta \approx 2/n^2$, we just need to lower bound $\frac{\text{Area}(S)}{\text{Vol(S)}}$. Fix a value of $\text{Vol}(S)$. How small can $\text{Area}(S)$ be? This is called an *isoperimetric inequality.*

  Theorem 4.27. *Let $L \subseteq \mathbb{R}^n$ be convex with diameter d . Let $S \subseteq L$ be such that $Vol(S) \leq Vol(L)/2$. Then, $Area(S) \geq \frac{1}{d} Vol(S)$.*

  If we look at a body L that is a very thin needle, we see that for S that cuts across half the needle the volume will be roughly $(d/2)$ times the area, so the estimate provided in the theorem is essentially optimal. The proof of the theorem is not hard but it does involve quite a bit of re-arranging to argue that the needle is indeed the "worst-case scenario", and we'll skip it.

  In our setting we have $B_\infty(0,1) \subseteq K \subseteq L \subseteq L' = (1+\delta\sqrt{n})L \subseteq (1+\delta\sqrt{n})B_\infty(0,n^2)$. So, our grid has sides of length at most $\lesssim \frac{2(1+\delta\sqrt{n})n^2}{\delta} = O(n^4)$, and the diameter is $O(n^4)$. Thus the isoperimetry theorem implies that our random walk will mix in polynomial time.

□

4.6 Expander Graphs

Suppose that we have a collection of points $V = \{1, ..., N\}$, where N is very large. Our goal is to uniformly sample points from V using as little space as possible. To do this, we place

a graph on V and select the points with a random walk. What types of graphs would be good for this procedure? We could throw in all edges — this will optimize the mixing time. But then at any step we need to choose amongst N possible neighbors, and this can be a complicated task; especially in situations such as we encountered in the last lecture, where determining if a point is a neighbor requires to do some computation (in that case, we had to make a call to the membership oracle, which could be expensive). To reduce the number of choices that the random walk has to make at each step, we want to minimize the degree. But we don't want to sacrifice the mixing time either. *Expander graphs* are often used because they reach the optimal tradeoff, achieving the best possible mixing time (or more precisely, the largest possible second eigenvalue) for a fixed degree.

4.6.1 Definitions of expanders

There are several definitions of expanders which are all more-or-less, but not quite, equivalent. The one that we will focus on is that of (two-sided) *spectral expanders*.

Definition 4.28. Given $d \in \mathbb{N}$ and $\gamma \in (0,1)$, a one-sided (resp. two-sided) (d, γ) spectral expander is a graph G such that:

- G is d-regular.
- $|\lambda_2(\bar{L}) - 1| \leq \gamma$ (resp. $\forall i \geq 2, |\lambda_i(\bar{L}) - 1| \leq \gamma$), where $\overline{L}$ is the normalized Laplacian of G.

Two-sided expanders have all their eigenvalues close to 1, except $\lambda_1 = 0$; one-sided expanders can also have larger eigenvalues, up to the maximum possible of 2 (and in particular they can be bipartite).

Remark 4.29. Using the mixing lemma from last lecture we find that the lazy random walk on expanders mixes fast:

$$\tau_\epsilon = O\Big(\frac{\log(\frac{n}{\epsilon})}{1-\gamma}\Big).$$

For the case of expanders (and especially two-sided expanders, which are guaranteed to not be bipartite) we will usually run a normal random walk with walk matrix $W = AD^{-1}$, rather than the lazy random walk with walk matrix $W = \mathbb{I} - \frac{1}{2}D^{1/2}\bar{L}D^{-1/2}$ that we considered previously.

There are two other common definitions of expanders:

Definition 4.30. A (d, γ_e) edge expander is a graph G such that:

- G is d-regular with degree d.
- $\phi(G) = \min\limits_{1 \leq |S| \leq \frac{n}{2}} \frac{|\partial S|}{d \cdot |S|} \geq \gamma_e$.

Definition 4.31. A (d, γ_v) vertex expander is a graph G such that:

- G is d-regular with degree d.
- $\sigma(G) = \min_{1 \le |S| \le \frac{n}{2}} \frac{|N(S)|}{|S|} \ge \gamma_v$, where $N(S)$ is the set of all neighbors of S lying outside of S. (Here $\sigma(G)$ is called the *vertex expansion* of G.)

The following relation holds between these two notions:

$$\frac{\gamma_v}{d} \le \gamma_e \le \gamma_v.$$

The first inequality follows from the fact that the number of edges entering S is at least the size of its neighborhood. The second inequality follows from the size of the neighborhood being at least the number of edges divided by the degree.

4.6.2 Limits on expansion

From this point on we will focus on two-sided spectral expanders. Given a target degree d, how small can we make γ?

Theorem 4.32. *For any d-regular graph,*

$$\gamma := \max_{i=2,\dots,n} |\lambda_i(\bar{L}) - 1| \ge 2(1 - o_n(1))\sqrt{\frac{1}{d} - \frac{1}{d^2}}$$
$$\sim 2\sqrt{\frac{1}{d}}.$$

We prove a slightly weaker version of this bound, which is missing the factor 2:

Proof. If A is the adjacency matrix of a graph G then for any $k \ge 1$ the (i, j)-th entry $(A^k)_{ij}$ counts the number of paths from i to j with length exactly k. So $(A^2)_{ii}$ is just the number of edges incident on the i-th vertex, i.e. its degree. Therefore $\mathrm{Tr}(A^2) = nd$. We also know that for a symmetric matrix,

$$\begin{aligned}
\mathrm{Tr}(A^2) &= \sum_{i=1}^{n} \lambda_i^2(A) \\
&= \sum_{i=1}^{n} \lambda_i^2(d(\mathbb{I} - \bar{L})) \\
&= d^2 \sum_{i=1}^{n} (1 - \lambda_i(\bar{L}))^2 \\
&\le d^2 + d^2(n-1)\gamma^2.
\end{aligned}$$

Therefore $d^2(n-1)\gamma^2 \geq nd - d^2$, i.e.

$$\gamma^2 \geq \frac{n}{d(n-1)} - \frac{1}{n-1}.$$

For large n the last term is very small, and we get the bound $\gamma \geq (1 - o_n(1))\sqrt{\frac{1}{d}}$, which is off by just a factor 2. □

Ramanujan graphs are graphs that achieve the optimal expansion for a given degree:

Definition 4.33. A Ramanujan graph is a d-regular graph such that

$$\gamma = 2\sqrt{\frac{1}{d} - \frac{1}{d^2}}.$$

4.6.3 Constructions of expanders

Do such good expanders as Ramanujan graphs even exist? The probabilitic method can be used to show that random d-regular graphs are good expanders with high probability:

Theorem 4.34 (Friedman)**.** *$\forall \epsilon > 0$, a random d-regular graph on n vertices satisfies:*

$$\mathbf{Pr}\left(\gamma \leq 2\sqrt{\tfrac{1}{d} - \tfrac{1}{d^2}} + \epsilon\right) = 1 - o_n(1).$$

Unfortunately random graphs are not so useful in practice because they are just that: random. In particular, working with a random graph requires to first compute the whole graph, then store it in memory and perform the random walk. But recall that for our typical applications of expanders we are thinking of working with potentially large graphs for which we'd like to be able to compute the neighborhood structure locally very efficiently. Such a construction was presented by Margulis; efficiency aside it is the first explicit construction of a good (meaning that both d and γ are constants independent of the size of the graph) family of expanders:

Example 4.35 (Margulis '73)**.** Take $V = \mathbb{Z}_m \times \mathbb{Z}_m$ for some integer m. For each vertex $(x, y) \in V$ connect it to the following eight vertices:

$$N((x,y)) = \begin{cases} (x, y \pm x) \\ (x, y \pm (x+1)) \\ (x \pm y, y) \\ (x \pm (y+1), y) \end{cases}$$

Theorem 4.36 (Margulis)**.** *The graph given above is a $(8, \gamma)$ two-sided spectral expander for some $\gamma < 1$ independent of m.*

The construction of the graph is very simple, but the proof of the theorem is very difficult. This construction also provides a very fast mixing time: it provides a way to mix a 2-dimensional $m \times m$ grid in time $O(\log m)$, whereas as we've seen the regular random walk would require time $O(m^2)$. This only requires us to double the degree and throw in a few "long-distance hops".

The first construction of explicit Ramanujan expanders was given by Lubotzky, Philips and Sarnak:

Theorem 4.37 (LPS '88)**.** *for every $n = p + 1$ where:*

- *p is a prime such that $p \equiv 1[4]$,*
- *$d = q^m + 1$, q prime, m integer,*

there exists a $(d, \gamma = 2\sqrt{\frac{1}{d} - \frac{1}{d^2}})$ spectral expander.

Very recent works by Marcus, Spielman and Srivastava give explicit constructions of *bipartite* expanders for every possible degree d and size n. It is still an open problem whether (non-bipartite) Ramanujan expanders exist for all possible degrees.

4.7 Derandomization

A couple lectures ago we saw a randomized algorithm that could efficiently solve a problem, volume estimation, that we also argued was too hard to solve deterministically. One might ask whether it is still possible to remove the randomness from such algorithms to obtain efficient deterministic algorithms. The process of reducing or eliminating randomness from an algorithm is called **derandomization**. The question of derandomizing all polynomial-time algorithms is the question as to whether **P** = **BPP**, where:

Definition 4.38. **P** is the class of problems that can be solved deterministically in polynomial time. **BPP** is the class of problems for which there is a randomized polynomial-time algorithm that makes the correct decision for every input with probability $\geq \frac{2}{3}$.

In the last lecture with volume estimation we seemed to show that for a specific problem in **BPP** it was impossible to have an equivalent deterministic algorithm — thus **P** $\neq$ **BPP**? However the impossibility result relied on the assumption that the only access to the convex set was through a membership/separation oracle. In "real life" it will in general be the case that other details of the problem are available, and may be used to construct a deterministic polynomial-time algorithm. This is what makes proving separations of complexity classes difficult!

4.7.1 Pseudorandom Generators

A popular approach for trying to show that **P** = **BPP** is constructing good pseudorandom generators (PRG).

Definition 4.39. A pseudorandom generator is a function $g : \{0,1\}^s \to \{0,1\}^n$, where $n \geq s$ and s is called the *seed length*. We say that a PRG g ϵ-fools a class of functions $\mathcal{C} \subseteq \{f : \{0,1\}^n \to \{0,1\}\}$ if $\forall f \in \mathcal{C}$:

$$\left| \mathop{\mathbf{Pr}}_{x \in \{0,1\}^n} \left[f(x) = 1 \right] - \mathop{\mathbf{Pr}}_{y \in \{0,1\}^s} \left[f(g(y)) = 1 \right] \right| \leq \epsilon,$$

where both probabilities are taken over a uniformly random choice of x, y respectively. (Intuitively, this means that from the point of view of any function $f \in \mathcal{C}$, the output of g looks random)

Here is how we might use this to prove that $\mathbf{P} = \mathbf{BPP}$. Let $\mathcal{C}$ be the class of all functions that can be computed by a polynomial time-randomized algorithm with success probability at least $2/3$. Such algorithms can be understood as functions f of two variables, the input x to the problem and the randomness r. Fixing all possible inputs x of a certain length gives us a large class of functions $\mathcal{C}'$ which are functions of the randomness only. If we design a PRG that ϵ-fools the class $\mathcal{C}'$ with $\epsilon < \frac{1}{3}$ and $s = O(\log n)$, then we could derandomize **BPP** by trying all $2^s = \text{poly}(n)$ possible seeds to our PRG and computing a good estimate of the probability that f would accept on any input.

Pseudorandom generators are hard to construct. Nisan and Wigderson in 94 proved the following "hardness vs randomness" trade-off. Here we state a strong form of their theorem that incorporates a worst-case to average-case reduction by Impagliazzo and Wigderson.

Theorem 4.40 (NW'94,IW'97)**.** *Suppose there is a language L in EXP and $\delta > 0$ such that for all n large enough, the minimal size of a Boolean circuit that computes L_n is at least $2^{\delta n}$. Then then there is a family of generators $g_m : \{0,1\}^{O(\log m)} \to \{0,1\}^m$ that are computable in* poly(m) *time and that $1/8$-fool the class of functions computable by circuits of size at most $2m$ (in particular, $\mathbf{P} = \mathbf{BPP}$).*

4.7.2 Expanders for derandomization

We will use expanders, not with the intent of producing a PRG, but to derandomize the following problem: Suppose that we have a **BPP** algorithm which given an input generates random bits in $\{0,1\}^r$ and has an error rate of $\frac{1}{100}$. If we run the algorithm t times and take the majority, then we can reduce the error to $\sim 2^t(\frac{1}{100})^{t/2}$. Doing so requires $t \cdot r$ bits. We will show that by using expanders, we can get a similar result using much fewer bits.

Theorem 4.41. *Given any* **BPP** *algorithm which requires r random bits and has error $\leq \frac{1}{100}$, we can construct an algorithm solving the same problem with error $\leq (\frac{2}{\sqrt{5}})^t$ and using only $r + 9t$ random bits.*

The idea for the algorithm is very simple. We fix a (d, γ) expander G on $V = \{0,1\}^r$ with $d \leq 400$ and $\gamma \leq \frac{1}{10}$. We haven't seen how to construct such a thing but I promise you that it exists, and it's not that hard to get. We then pick a random vertex to start from using r bits, and for each of t iterations we perform a random walk to a neighbor using $\log_2(400) \approx 9$

bits. For each of the vertices traversed we run the algorithm with the corresponding bits and take the majority outcome at the end.

Note that we do not need to actually compute the whole graph (which would have size exponential in r); we only need to be able to access a small number of vertices and their neighbors. The hope is that running the **BPP** algorithm on the highly correlated pseudo-random bits generated by this procedure will generate similar results to uncorrelated random bits. This is not trivial, and in particular it does not follow from the analysis of the mixing time given in the previous lecture. That analysis only shows that mixing happens in $O(\log 2^r) = O(r)$ steps, but here t could be much smaller than r.

Proof. Given an input x, we know that for any set of random bits y,

$$\mathop{\mathbf{Pr}}_{y\in\{0,1\}^r}\big(\text{Alg fails on input } x \text{ and randomness } y\big) \le \frac{1}{100}.$$

Fix an input x, and let $X = \{y : \text{Alg fails on } y\}$, $Y = \{0,1\}^r \setminus X$, $v_0, v_1, \ldots, v_t$ the vertices selected from the random walk and $S = \{i : v_i \in X\}$. Note that X is at most a fraction $\frac{1}{100}$ of the graph and that since we are taking majority at the end, $\mathbf{Pr}(fail) = \mathbf{Pr}(|S| \ge t/2)$. We will use a regular "non-lazy" random walk, as there is no point in staying at a vertex. This gives us a random walk matrix $W = AD^{-1} = \frac{1}{d}A$ since the graph is d-regular.

Let D_X be the diagonal matrix where entry i is 1 if $i \in X$ and 0 otherwise, and D_Y the diagonal matrix where entry i is 1 if $i \in Y$ and 0 otherwise. So $D_X + D_Y = \mathbb{I}$. Given a certain set of designated walk steps $R \subseteq \{0, \ldots, t\}$ the probability that all such steps are errors and all other steps are correct is:

$$\mathbf{Pr}(R = S) = \mathbf{1}^T D_t W \cdots D_1 W D_0 \frac{\mathbf{1}}{n}, \qquad \text{where} \qquad D_i = \begin{cases} D_X \text{ if } i \in R \\ D_Y \text{ if } i \notin R \end{cases}.$$

So for any fixed R,

$$\begin{aligned}
\mathbf{Pr}(R = S) &\le \|\mathbf{1}\|_2 \cdot \|D_t W\| \cdots \|D_1 W\| \cdot \|D_0 W\| \Big\|\frac{\mathbf{1}}{n}\Big\|_2 \\
&= \sqrt{n} \cdot \|D_X W\|^{|R|} \cdot \|D_Y W\|^{t-|R|} \frac{1}{\sqrt{n}} \\
&\le \|D_X W\|^{|R|},
\end{aligned}$$

where the last inequality follows since $\|W\| = \|\frac{A}{d}\| \le 1 \to \|D_Y W\| \le 1$. Now it remains to bound $\|D_X W\|$. Consider any vector x. We can use the decomposition $x = \alpha\mathbf{1} + y$, $y \perp \mathbf{1}$ (these are not the x and y from the original problem description!). Then

$$\|x\|^2 = \alpha^2 n + \|y\|^2,$$

and

$$D_X W x = \alpha D_X W \mathbf{1} + D_X W y.$$

To bound the first term, use that $W\mathbf{1} = \mathbf{1}$ and so

$$\|\alpha D_X W \mathbf{1}\| = |\alpha|\sqrt{|X|} \leq \frac{\|x\|}{\sqrt{n}}\sqrt{|X|} \leq \frac{\|x\|}{10}.$$

To bound the second term,

$$\|D_X W y\| \leq \|Wy\| \leq \frac{1}{10}\|y\| \leq \frac{\|x\|}{10},$$

where the second inequality follows since $y \perp \mathbf{1}$. Thus we have shown that $\|D_X W\| \leq \frac{1}{5}$. Finally, putting everything together,

$$\begin{aligned}
\mathbf{Pr}(fail) &= \mathbf{Pr}\left(|S| \geq \frac{t}{2}\right) \\
&= \sum_{R:|R|\geq\frac{t}{2}} \mathbf{Pr}[R = S] \\
&\leq \sum_{R:|R|\geq\frac{t}{2}} \|D_X W\|^{|R|} \\
&\leq 2^t (\frac{1}{5})^{\frac{t}{2}}.
\end{aligned}$$

□

4.7.3 Polynomial Identity Testing

There are relatively few candidates problems known that could provide a separation between **P** and **BPP**. One example is volume estimation, that we saw a couple lectures ago. Another example is, given an integer n (in unary), to generate a prime number $2^n \leq p \leq 2^{2n}$: while thanks to the famous AKS algorithm it is possible to *test* if a given number is prime in polynomial time, it remains unknown how to (deterministically) *find* one. (Thanks to the prime number theorem there is an easy randomized algorithm: generate a random number and test if it is prime; this should work with probability $1 - O(1/n)$.) A third example is the problem of factoring a univariate polynomial over a finite field into irreducible factors. There are efficient randomized algorithms for this problem (e.g. the Cantor-Zassenhaus algorithm), but no worst-case polynomial time algorithm is known (although there are deterministic algorithms, such as Shoup's algorithm, that take exponential time for some polynomials, but have polynomial expected running time when the polynomial is chosen at random).

Here we focus on a fourth example, *polynomial identity testing* (PIT):

Definition 4.42 (Polynomial Identity Testing)**.** Given a (large) finite field $\mathbb{F}$ and an n-variate polynomial $p \in \mathbb{F}[x_1, ..., x_n]$ provided as an arithmetic circuit (eg. $(x_1+3x_2-x_3)(3x_1+x_4-1)$), determine whether $p \equiv 0$, i.e. all coefficients of p when fully expanded are zero.

To solve the above problem we could simply multiply all the terms out and check whether the final coefficient of every term is 0, but that could take exponential time in the size of the circuit specifying p. No known deterministic algorithm can solve PIT in polynomial time, but there is a simple randomized algorithm that can.

Lemma 4.43 (Schwartz-Zippel)**.** *Given a non-zero $p \in \mathbb{F}[x_1, ..., x_n]$ with degree$(p) \leq d$, let $S \subseteq \mathbb{F}$ and $(s_1, ...s_n)$ be n points selected independently and uniformly at random from S. Then:*

$$\mathbf{Pr}\left(p(s_1, ..., s_n) = 0\right) \leq \tfrac{d}{|S|}.$$

Proof. The proof is by induction on n. For the base case, $n = 1$, p is a polynomial in one variable and thus has at most d roots. Hence $\mathbf{Pr}(p(s_1) = 0) \leq d/|S|$.

For the inductive step, we let k be the largest degree of x_1 in p and write

$$p(s_1, \ldots, s_n) = p_1(s_2, \ldots, s_n)s_1^k + p_2(s_1, \ldots, s_n),$$

where the maximum degree of $p_1(s_2, \ldots, s_n)$ is at most $d - k$ and the maximum degree of s_1 in p_2 is strictly less than k. For the purposes of analysis, we can assume that $s_2, \ldots, s_n$ are chosen first. Then, we let $\mathcal{E}$ be the event that $p_1(s_2, \ldots, s_n) = 0$. There are two cases:

- *Case 1: $\mathcal{E}$ happens.* From the induction hypothesis applied to M (a polynomial in $n - 1$ variables), we know that $\mathbf{Pr}(\mathcal{E}) \leq (d - k)/|S|$.
- *Case 2: $\mathcal{E}$ does not happen.* In this case, we let p' be the polynomial in one variable x_1 that remains after $x_2 = s_2, \ldots, x_n = s_n$ are substituted in $p(x_1, \ldots, x_n)$. Since $p_1(s_2, \ldots, s_n) \neq 0$, the coefficient of of x_1^k is non-zero, so p' is a non-zero polynomial of degree k in one variable. It thus has at most k roots, so $\mathbf{Pr}(p'(r_1) = p(r_1, \ldots, r_n) = 0|\neg\mathcal{E}) \leq k/|S|$.

Putting the two cases together, we have

$$\begin{aligned}\mathbf{Pr}\left(p(s_1, \ldots, s_n) = 0\right) &= \mathbf{Pr}\left(p(s_1, \ldots, s_n) = 0|\mathcal{E}\right)\mathbf{Pr}\left(\mathcal{E}\right) \\ &\quad + \mathbf{Pr}\left(p(s_1, \ldots, s_n) = 0|\neg\mathcal{E}\right)\mathbf{Pr}\left(\neg\mathcal{E}\right) \\ &\leq (d - k)/|S| + k/|S| = d/|S|.\end{aligned}$$

□

Thus, if we take the set S to have cardinality at least twice the degree of our polynomial, we can bound the probability of error by 1/2. This can be reduced to any desired small number by repeated trials, as usual. Note that the algorithms also works over finite fields, *provided* the field size is larger than the degree of the polynomial. Otherwise, the algorithm could not possibly work: for example, the polynomial $p(x) = x^3 - x$ is not the zero polynomial, but it evaluates to 0 on every point in $\mathbb{F}_3$.

Is there an efficient *deterministic* algorithm for identity testing? A rather devastating negative result was proved by Kabanets and Impagliazz, who showed that if there exists a deterministic polynomial time algorithm for checking polynomial identities, then either:

- NEXP does not have polynomial size circuits; or
- The permanent cannot be computed by polynomial-size arithmetic circuits.

This means that an efficient derandomization of the above Schwartz-Zippel algorithm (or indeed, any efficient deterministic algorithm for identity testing) would necessarily entail a major breakthrough in complexity theory.

Using the Schwartz-Zippel algorithm

We give an application of the Schwartz-Zippel lemma to bipartite matching. Although bipartite matching is easy to solve in deterministic polynomial time using flow techniques, it remains an important open question whether there exists an efficient *parallel* deterministic algorithm for this problem.

Bipartite matching is the following problem: Given a bipartite graph $G = (V_1, V_2, E)$, where $|V_1| = |V_2| = n$, and all edges connect vertices in V_1 to vertices in V_2, does G contain a perfect matching, that is, a subset of exactly n edges such that each vertex is contained in exactly one edge?

To obtain an efficient randomized algorithm for this problem, we begin with a definition:

Definition 4.44 (Tutte matrix)**.** The Tutte matrix A_G corresponding to the graph G is the $n \times n$ matrix $[a_{ij}]$ such that a_{ij} is a variable x_{ij} if $(i, j) \in E$, and 0 otherwise.

Claim 4.45. *G contains a perfect matching if and only if $det(A_G) \neq 0$.*

Proof. By definition, $\det(A_G) = \sum_\sigma \operatorname{sgn}(\sigma) \prod_{i=1}^n a_{i\sigma(i)}$, where the sum is over all permutations σ of $\{1, \ldots, n\}$. Note that each monomial in this sum corresponds to a *possible* perfect matching in G, and that the monomial will be non-zero if and only if the corresponding matching is present in G. Moreover, every pair of monomials differs in at least one (actually, at least two) variables, so there can be no cancellations between monomials. This implies that $\det(A_G) \not\equiv 0$ iff G contains a perfect matching. □

The above Claim immediately yields an efficient algorithm for testing whether G contains a perfect matching: simply run the Schwartz-Zippel algorithm on $\det(A_G)$, which is a polynomial in n^2 variables of degree n. Note that the determinant can be computed in $O(n^3)$ time by Gaussian elimination. Moreover, the algorithm can be efficiently parallelized using the standard fact that an $n \times n$ determinant can be computed in $O(\log^2 n)$ time on $O(n^{3.5})$ processors.

Exercise 5. Generalize the above to a non-bipartite graph G. For this, you will need the skew-symmetric matrix $A_G = [a_{ij}]$ defined as:

$$a_{ij} = \begin{cases} x_{ij} & \text{if } (i,j) \in E \text{ and } i < j; \\ -x_{ij} & \text{if } (i,j) \in E \text{ and } i > j; \\ 0 & \text{otherwise.} \end{cases}$$

[Hint: You should show that the above Claim still holds, with this modified definition of A_G. This requires a bit more care than in the bipartite case, because the monomials in $\det(A_G)$ do not necessarily correspond to perfect matchings, but to cycle covers in G. In this case some cancellations will occur.]