Today we will study electrical networks and their connections to random walk and graph sparsification. First we will formulate the problem (as a Laplacian system) and see some basic properties (e.g. energy minimization). Then we study its connection to hitting times of random walk in undirected graphs. Finally we will see a surprising connection to graph sparsification, that will be useful next week in designing fast Laplacian solvers.

Overview

We are on the way of a near linear time solver for Laplacian systems. In the last two weeks we studied fast iterative solvers of linear equations, including the conjugate gradient method and its convergence analysis and the preconditioning technique. We then focused on solving Laplacian systems, where combinatorial techniques can be used to design preconditioners with provable performance guarantee (using low stretch spanning trees to obtain an $O(m^{4/3})$-time solver for Laplacian systems).

We have almost all the ingredients for an $O(m)$-time solver for Laplacian systems. The final ingredient is clearly related to electrical networks, the use of effective resistances for graph sparsification, and this would imply efficient preconditioners as we will see in the next week.

Besides being a main ingredient for the Laplacian solvers, the study of electrical networks is of interest on its own. We will see the interesting connection between effective resistances and hitting times of random walk, that helps answer some basic questions of random walk in undirected graphs. As we will see soon, computing effective resistances can be formulated as solving a Laplacian system, you can also think of this as an application of the Laplacian solver to the study of random walk.

Next week we will use the graph sparsification result to obtain a near linear time solver. In the last week we will use electric flow and the Laplacian solver to design a fast approximation algorithm for the classical maximum flow problem.
**Electric Flow**

We think of an undirected graph as an electrical network, where each edge $e$ is a resistor with resistance $R_e$.

The flow of electric current is governed by two rules:

1. **Kirchhoff's law (flow conservation law):** The sum of the currents entering a node is equal to the sum of currents leaving it.

2. **Ohm's law (potential flow law):** The potential drop across a resistor is equal to the current flowing over the resistor times the resistance.

For example, consider this network:

If one ampere is injected into $a$ and one ampere is removed from $t$, then the voltages at the nodes and the currents on the resistors are shown in the figure on the right.

**Notation**

Let's write a matrix formulation of the problem.

Let $G = (V,E)$ be the underlying undirected graph.

Let $V \in \mathbb{R}^V$ be the vector of potentials at vertices.

Let $i(a,b) \in \mathbb{R}$ be the current flowing from vertex $a$ to vertex $b$ for an edge $(a,b)$. As this is a directed quantity, we define $i(b,a) = -i(a,b)$.

Let $l \in \mathbb{R}^E$ be the vector of currents flowing over the edges, where each edge $e = (a,b)$ appears once as $i(a,b)$ where $a < b$.

Let $w_e = 1/R_e$ be the "conductance" of the edge $e$.

**Matrix Formulation**

The Ohm's law states that $i(a,b) = \frac{v(a) - v(b)}{R_{ab}} = w_{ab} \cdot (v(a) - v(b))$.

The Kirchhoff's law states that $\sum_{b \in N(a), b \neq a} i(a,b) = i_{ext}(a)$, where $i_{ext}(a)$ denotes the external current entering the network through the node $a$, so it is a positive number if $a$ is a source and a
negative number if \( a \) is a sink and zero otherwise.

By Ohm's law, \( \sum_{b \in \text{in}(a)} i(a,b) \cdot w_{a,b} \cdot (v(a) - v(b)) = -d(a) \cdot v(a) - \sum_{b \in \text{out}(a)} w_{a,b} \cdot v(b) \), where \( d(a) = \sum_{b \in \text{in}(a)} w_{a,b} \) is the weighted degree of \( a \).

Then this is just equivalent to \( L_G \mathbf{v} = \mathbf{i}_{\text{ext}} \), where \( L_G \) is the weighted Laplacian of \( G \) and \( \mathbf{i}_{\text{ext}} \) is the vector of external currents at the vertices.

### Computing Voltages

Therefore, if we can solve a Laplacian system quickly, then we can compute the voltages (and thus the currents) quickly.

Notice that \( L_G \) is not of full rank. Assume without loss of generality that \( G \) is connected.

Then we know that nullspace \( (L_G)^T \) is nontrivial. Last time we also see that range \( L_G \) is isomorphic to nullspace \( (L_G)^T \).

In particular, there is a solution to \( L_G \mathbf{v} = \mathbf{i}_{\text{ext}} \) if and only if \( \mathbf{i}_{\text{ext}} \perp \mathbb{1} \), which should be clear to our problem as the total external currents injecting into the network should be equal to the total external currents removing from the network.

Let \( \lambda_1 < \lambda_2 < \ldots < \lambda_n \) be the eigenvalues of \( L_G \) with corresponding eigenvectors \( \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \).

Then \( L_G = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^T \). Recall from last time that we define the pseudo-inverse \( L_G^+ \) (usually written as \( L_G^{-1} \)) as \( L_G^{-1} = \sum_{i=1}^{n} \frac{1}{\lambda_i} \mathbf{v}_i \mathbf{v}_i^T \). If \( \mathbf{x} \perp \mathbb{1} \), then \( \mathbf{y} = L_G \mathbf{x} \) is the unique solution to \( L_G \mathbf{y} = \mathbf{x} \) that also satisfies \( \mathbf{y} \perp \mathbb{1} \), and the set of all solutions is \( \mathbf{y} + c^T \mathbb{1} \) for \( c \in \mathbb{R} \).

### Computing Currents

Once we have computed the voltages, then it is easy to compute the currents.

Let us write down a matrix formulation for our discussions later.

Let \( B \) be a \( m \times n \) matrix whose row are indexed by the edges and the columns are indexed by the vertices, and the row corresponding to the edge \( e = (a,b) \) with \( a < b \) is \( (X_a, X_b)^T \), where \( X_a \) is the characteristic vector with one in the \( a \)-th entry and zero otherwise.

Let \( W \) be the \( n \times n \) diagonal matrix where \( W_{e,e} = w_{e} \) is the weight of edge \( e \).

Then \( \mathbf{i} = W B \mathbf{v} \).

Notice that \( L_G = \sum_{e \in \text{edge}} w_e (X_a-X_b)(X_b-X_a)^T = B^T W B \). So, \( \mathbf{i}_{\text{ext}} = L_G \mathbf{v} = B^T W B \mathbf{v} = B^T \mathbf{i} \), which can also be checked directly from the definition.
Effective Resistance

The effective resistance between vertices $a$ and $b$ is defined as $v(a) - v(b)$ when one ampere is injected into $a$ and removed from $b$. You can think of it as the resistance between $a$ and $b$ given by the whole network. We denote it by $R_{ab}(a,b)$.

To compute $R_{ab}(a,b)$, first we compute the voltages when one ampere is injected into $a$ and removed from $b$. By the matrix formulation, this is the solution of $Lqv = (x_a - x_b)$, which is given by $v = L_q^{-1}(x_a - x_b)$. Then $R_{ab}(a,b)$ is just $(x_a - x_b)^T L_q^{-1}(x_a - x_b)$.

So, once we have $L_q$, we can compute $R_{ab}(a,b)$ for all $a,b$ easily.

Energy

Recall from physics that the energy dissipated in a resistor network with currents $i(x,y) v_{x,y}$ is:

$$E(i) \overset{\text{def}}{=} i^T R_i = \sum_{(x,y) \in \mathcal{E}} i(x,y) R_{x,y} = \sum_{(x,y) \in \mathcal{E}} \frac{1}{R_{x,y}} (v(x) - v(y))^2 = \sum_{(x,y) \in \mathcal{E}} W_{x,y} (v(x) - v(y))^2 = v^T L_q v,$$

where $R$ is the (non-diagonal) matrix where $R_{x,y} = \frac{1}{R_{x,y}}$.

Intuitively, if we think of the whole network as one resistor from $a$ to $b$, then

$$R_{ab}(a,b) = (v(a) - v(b)) / i(a,b) = E(i)$$

if one unit of current is sent from $a$ to $b$.

This can be proved formally as $R_{ab}(a,b) = (x_a - x_b)^T L_q^{-1} (x_a - x_b) = (L_q v)^T L_q^{-1} (L_q v) = v^T L_q^{-1} L_q v = v^T L_q v = E(i)$,

as it is easy to verify that $L_q L_q^{-1} = L_q$.

Energy Minimization

The electric flow from $s$ to $t$ is the one that minimizes the energy.

Let $j$ be one unit of flow from $s$ to $t$, satisfying the flow conservation rule at every vertex.

Define its energy to be $E(j) = j^T R j = \frac{1}{2} (x_s - x_t)^2$.

Theorem (Thompson's Principle) $R_{ab}(s,t) \leq E(j)$.

Proof. Let $i$ be the electrical flow of one unit from $s$ to $t$, and $v$ be the corresponding voltages.

Consider $C=j-i$.

As both $i$ and $j$ satisfy flow conservation constraints, we have $B^i = B^j = (x_s - x_t)$ as the $v$-th entry of $B^i$ is $\sum_{(w,v) \in \mathcal{E}} (-i(w,v)) + \sum_{(v,w) \in \mathcal{E}} i(v,w) = \sum_{w \in \mathcal{V}} i(v,w) = i_{xt}(v)$.

Therefore, $B^c = B^i - B^j = 0$, and hence $\sum_{w \in \mathcal{V}} c(v,w) = 0$ for all $v$.

$$E(i) = \sum_i j(i,a,b) r_{a,b} = \sum_i (i(a,b) + c(a,b))^2 r_{a,b}$$

L12 Page 4
\[ E(i) = \sum_{a \neq b} j(a,b) \cdot r_{a,b} = \sum_{a \neq b} (i(a,b) + c(a,b))^2 \cdot r_{a,b} \]

\[ = \sum_{a \neq b} i(a,b) \cdot r_{a,b} + 2 \sum_{a \neq b} i(a,b) \cdot c(a,b) \cdot r_{a,b} + \sum_{a \neq b} c(a,b) \cdot r_{a,b}. \]

Observe that the first term is \( E(i) \), and the last term is positive if \( i \neq j \).

Hence we will complete the proof once we show that \( \sum_{a \neq b} i(a,b) \cdot c(a,b) \cdot r_{a,b} = 0. \)

To see this, \( \sum_{a \neq b} i(a,b) \cdot c(a,b) \cdot r_{a,b} = \sum_{a \neq b} (v(a) - v(b)) \cdot c(a,b) \) (by Ohm's law)

\[ = \sum_{a \neq b} (v(a) \cdot c(a,b) + v(b) \cdot c(b,a)) \]

\[ = \sum_{a \neq b} v(a) \sum_{b \neq a} c(a,b) = 0. \]

---

**Effective Resistance as Distance**

Let us try to get some intuition about the effective resistances.

The Rayleigh monotonicity principle says that the effective resistance cannot decrease if we increase the resistance of some edge.

**Theorem** (Rayleigh's Monotonicity Principle) Let \( r' \geq r \) be the resistances. Then \( E_r(i) \geq E_r(i) \),

where \( E_r(i) \) denotes the energy of flow \( i \) under the resistances \( r \).

**Proof** Let \( i \) and \( i' \) be the electric flow under resistances \( r \) and \( r' \) respectively.

Then \( E_r(i') \geq E_r(i) \) as \( r' \geq r \)

\[ \geq E_r(i) \] by the Thompson's principle.

Intuitively, if there is a short path between \( s \) and \( t \), then the effective resistance between \( s \) and \( t \) is small. Also, if there are many paths between \( s \) and \( t \), then the effective resistance between \( s \) and \( t \) is smaller. One can use the Rayleigh's monotonicity principle to give a bound on the effective resistance.

**Claim** If there are \( k \) edge-disjoint paths from \( s \) to \( t \), each of length at most \( l \).

Then \( R(s,t) \leq l/k \), assuming the graph is unweighted.

**Proof** Increase the resistances of all other edges to infinity. Then the effective resistance of the resulting graph is at most \( l/k \) by direct calculation. By monotonicity the effective resistance in the original network could not be larger than that.

Effective resistances provide an alternative way to measure the distance of two nodes in a graph.
Sometimes more useful than the traditional shortest path distance. For instance, one could use the effective resistances as distances to identify clusters in a social network.

Actually, effective resistances satisfy the triangle inequality.

**Claim** \( \text{Reff}(a,b) + \text{Reff}(b,c) \geq \text{Reff}(a,c) \).

**Proof** Let \( V_{a,b}, V_{a,c}, V_{b,c} \) be the voltages when one unit of current is sent from \( a \rightarrow b \), \( a \rightarrow c \), \( b \rightarrow c \), respectively.

Then \( V_{a,b} = L^{-1}_a (X_a - X_b) \), \( V_{a,c} = L^{-1}_a (X_a - X_c) \) and \( V_{b,c} = L^{-1}_b (X_b - X_c) \).

So \( V_{a,b} + V_{b,c} = V_{a,c} \).

\( \text{Reff}(a,c) = (X_a - X_c)^T V_{a,c} = (X_a - X_c)^T (V_{a,b} + (X_a - X_c) V_{b,c}) \).

Note that \( (X_a - X_c)^T V_{a,b} = V_{a,b}(a) - V_{a,b}(c) \leq V_{a,b}(a) - V_{a,b}(b) \) as \( V_{a,b}(c) \leq V_{a,b}(a) \) for all \( c \in V \).

Similarly, the second term is at most \( \text{Reff}(b,c) \), and hence the claim follows.

In the following we will talk about the connection between effective resistances and hitting times, which will give even more intuitions about using effective resistances as distances.

---

**Random Walk in Undirected Graphs** [3]

In week 6 we studied random walks in undirected graphs, and answered two basic questions, in what graphs is there a limiting distribution and what is the convergence rate to the limiting distribution if one exists.

Now we can answer the remaining two questions about hitting time and cover time using effective resistances.

One can also think of it as an application of Laplacian solvers to compute hitting time and cover time approximately.

Recall that the hitting time from \( a \rightarrow b \) is the expected number of steps to reach \( b \) if the random walk starts from \( a \), denoted by \( h_{ab} \).

The cover time is the expected number of steps to reach every vertex at least once.

The commute time, denoted by \( C_{ab} \), is defined as \( h_{ab} + h_{ba} \).

**Theorem** \( C_{st} = \text{Reff}(s,t) \).

**Proof** Let \( v \in V - t \). Then \( h_{vt} = \frac{1}{\w_{vv} \w_{vt}} (s + h_{vt}) \), with \( h_{vt} = 0 \).
This is equivalent to \( d(u) = \Phi_u - \Phi_t = \sum_{w \in \mathcal{E}} (h_{w^+} - h_{w^-}) \) for \( u \in V - T \).

Observe that this is very similar to a Laplacian system of linear equations.

Let \( \Phi_u \) be the voltage at \( u \) with \( \Phi_t = 0 \), when \( d(u) \) units of currents are injected from \( u \in V - T \) and \( -2m \) units of current are removed from \( t \).

Then the values \( \Phi_u \) and \( h_u \) would satisfy the same equation.

This is because, for \( u \in V - T \), \( d(u) = \sum_{w \in \mathcal{E}} (\Phi_u - \Phi_t) \) by Ohm's law.

Let \( i_t \) be the vector of the external currents with \( i_t(u) = d(u) \) for \( u \in V - T \) and \( i_t(t) = -2m \). Let \( \Phi_t \) be the vector with \( \Phi_t(u) = \Phi_u \).

Then the values \( \Phi_u \) satisfy the Laplacian system \( J_T \Phi_u = i_t \) with \( \Phi_t = 0 \).

We know that the set of solution to this Laplacian system is \( \{ J_T^{-1} \Phi_t + c I \mid c \in \mathbb{R} \} \).

There is a unique solution with \( \Phi_t = 0 \), hence we must have \( h_u = \Phi_u \) for all \( u \).

Let \( i_s \) be the vector of external currents with \( i_s(u) = d(u) \) if \( u \in V - S \) and \( i_s(u) = -2m \) otherwise.

Then, as above, let \( h_s \) be the hitting time vector with \( h_s(u) = h_u \) and \( h_s(t) = h_t = 0 \).

Then \( h_s \) is the unique solution to \( J_T h_s = i_s \) with \( h_s = 0 \).

Now, \( L_T \left( h_t^+ - h_t^- \right) = i_t^+ - i_t^- = 2m (X_t^+ - X_t^-) \), and so \( \langle h_t^+ - h_t^- \rangle / 2m = L_T^{-1} (X_t^+ - X_t^-) \).

So, \( \frac{1}{2m} \langle h_t^+ - h_t^- \rangle \) is a voltage vector when \( 2m \) amperes are sent from \( s \) to \( t \).

\[
\text{Reff}(s, t) = \left\langle X_t^+ - X_t^- \right\rangle \left( \frac{1}{2m} \left( h_t^+ - h_t^- \right) \right) = \frac{1}{2m} \left( h_t^+ + h_t^- \right) = \frac{1}{2m} (h_{st} + h_{ts}) = \frac{1}{2m} C_{st} \]

Using this connection, we can use it to give bounds on the commute time and cover time.

**Corollary.** For any edge \( uv \), \( C_{uv} \leq 2m \).

**Proof.** The effective resistance between \( u \) and \( v \) is at most one, by Ohm's law. 

**Theorem.** The cover time of an undirected graph is at most \( 2m(n-1) \).

**Proof.** Let \( T \) be a spanning tree of \( G \).

Consider a walk that goes through \( T \) where each edge in \( T \) is transversed once in each direction.

Then this is a walk that visits every vertex at least once.

So the cover time of \( G \) is bounded by the expected length of this walk, which is at most \( \sum_{e \in T} (h_{w^+} + h_{w^-}) = \sum_{e \in T} C_e \leq 2m(n-1) \).

For the complete graph with \( n \) vertices, the cover time is \( O(n \log n) \) (coupon collector problem), but
the above bound gives only $O(n^3)$. The following is a tighter estimate of the cover time.

**Theorem.** Let $R(G) = \max_{u,v} R_{\text{eff}}(u,v)$. Then $mR(G) \leq \text{cover time} \leq 2^3 mR(G) \log n + n$.

**Proof.** Let $R(G) \geq R_{\text{eff}}(u,v)$. Then $2mR_{\text{eff}}(u,v) = C_{uv} + h_{uv} + h_{vu}$.

So the cover time is at least $\max \{ h_{uv}, h_{vu} \} \geq C_{uv}/2 = mR_{\text{eff}}(u,v)$, hence the lower bound.

For the upper bound, since the maximum hitting time is at most $2mR(G)$, regardless of the starting vertex.

So, a vertex is not covered after $2^3 mR(G)$ steps is at most $1/e^3$ by the Markov's inequality.

If the random walk runs for $2^3 mR(G) \ln n$ steps, then a vertex is not covered with probability at most $1/e^3$.

By the union bound, some vertex is not covered after $2^3 mR(G) \ln n$ is of probability at most $1/n^3$.

When this happens, we just use the bound that the cover time is at most $n^3$.

Then the cover time is at most $2^3 mR(G) \ln n + \left(\frac{1}{e^3}\right)^{n^3} \leq 2^3 mR(G) \ln n + n$.

---

**Spectral Sparsifier [4]**

We say a graph $H$ is an $\varepsilon$-approximation of a graph $G$ if $\varepsilon(I-G) \leq H \leq (1+\varepsilon)G$, i.e. $(1-\varepsilon)I - L_G \leq L_H \leq (1+\varepsilon)I - L_G$ for all $x \in \mathbb{R}^n$.

Our goal is: for any $G$, find a sparse $H$ that is an $\varepsilon$-approximation for a small $\varepsilon$.

Moreover, we want such an $H$ to be constructed efficiently.

To see that it is not an easy task, just put $x = 1_G$, then the requirement for this particular $x$ is $(1-\varepsilon)w_G(1_G) \leq w_H(1_G) \leq (1+\varepsilon)w_G(1_G)$. That is, the weight of each cut should be very close.

It is not clear such a sparsifier should exist at all, but it turns out that good sparsifiers exist.

**Theorem.** Given $G$ and $0 < \varepsilon \leq 1$, there exists $H$ with $O(n \log n / \varepsilon^2)$ edges such that $H$ is an $\varepsilon$-approximation of $G$. Furthermore, such an $H$ can be constructed in nearly linear time.

One immediate application of this theorem is to find an approximate minimum $s$-$t$ cut:

Given a graph $G$ and $s \to t \in V(G)$, first we construct an $\varepsilon$-approximation $H$ of $G$.

Then find an $s$-$t$ min-cut in $H$ is $O(mn) = O(n^2 \log n)$ time.

Then this cut is an $(1+\varepsilon)$-approximate min $s$-$t$ cut in $G$, and the running time is $O(n^2 \log n)$.

---

L12 Page 8
Then this cut is an $(1 + \epsilon)$-approximate min s-t cut in $G$, and the running time is $O(n \log n)$.

Another application is to construct efficient preconditioner for solving Laplacian systems, and we will discuss in details next week.

**Algorithm**

Consider the following general random sampling algorithm, which was used in previous graph sparsification algos.

Initially $H$ is the empty graph.

For $1 \leq i \leq q$:

- Pick a random edge $e$ with probability $p_e$.
- Increase the weight of $e$ in $H$ by $\frac{w_e}{q \cdot p_e}$.

That's all, but we have not specified the probability distribution $p_e$ and $q$.

It turns out that setting $p_e = \frac{w_e \cdot R_{e}}{\sum w_e \cdot R_{e}}$ and $q = O(n \log(n/\epsilon))$ will work, where $R_{e}$ denotes the effective resistance of $e$, i.e., if $e = (u,v)$ then $R_{e} = R_{eff}(u,v)$.

This algorithm is quite magical, but we will try to explain it in the most logical way.

**Setup**

We can think of the algorithm as scaling the edge weights (while setting most to zero).

Let $S$ be the norm diagonal matrix with $S(e,e) = \frac{w_e}{\sum w_e \cdot R_{e}}$, where $w_e$ is the weight in $H$.

Then $L_G = B^T W B$ and $L_H = B^T W^H S W^H B$, since $\tilde{w}_e = S(e,e) w_e$.

**Proof Outline**

The proof consists of three main steps. The first step is to reduce the problem to bounding the norm of a matrix. The second step is to define an appropriate projection matrix for the first step. The third step is to use existing results to bound the matrix norm.
Reducing to Matrix Norm

Our goal is to prove that \((1-\epsilon)x^T L x \leq x^T L x \leq (1+\epsilon)x^T L x\).

When \(x \in \text{nullspace}(L)\), then \(x^T L x = x^T L x = 0\), and there is nothing to prove.

So focus on \(x \notin \text{nullspace}(L) = \text{nullspace}(W^B J)\).

The goal is the same as \(\epsilon \geq \sup_{x \in \mathbb{R}^n, x^T B x > 0} \frac{|x^T L x - x^T L x|}{x^T L x} = \sup_{x \in \mathbb{R}^n, x^T B x > 0} \frac{|x^T B x^T S B x - x^T B W B x|}{x^T B W B x}\)

\[= \sup_{y \in \text{im}(W^B J), y \neq 0} \frac{|y^T (S-I) y|}{y^T y}\]

This form is like bounding the matrix norm of \((S-I)\), but not quite, as there is a restriction on an \(n\)-dimensional subspace. Just bounding \(\|S-I\|_2\) is too weak.

The idea is to project an \(n\)-dimensional vector into the subspace \(\text{im}(W^B J)\).

We can do it by defining a projection matrix \(\Pi\) such that if \(y \in \text{im}(W^B J)\) then \(\Pi y = y\), and if \(y \notin \text{im}(W^B J)\) then \(\Pi y \in \text{im}(W^B J)\).

Suppose such a projection matrix \(\Pi\) exists.

Then the above goal is equivalent to \(\epsilon \geq \sup_{y \in \text{im}(W^B J), y \neq 0} \frac{|y^T \Pi^T (S-I) \Pi y|}{y^T y}\)

\[= \sup_{y \in \text{im}(W^B J), y \neq 0} \frac{|y^T \Pi^T (S-I) \Pi y|}{y^T y}\]

Since \(\text{im}(\Pi) = \text{im}(W^B J)\), and thus we can remove the constraint \(y \in \text{im}(W^B J)\).

What we gain is that the problem is reduced to bounding \(\|\Pi^T S \Pi - \Pi^T \Pi\|_2\).

This is a more standard problem and there may be some previous work on it.

---

Projection Matrix

Given a \(m \times n\) matrix \(A\) with \(m \leq n\), we would like to construct a matrix \(\Pi\) such that if \(w \in \text{im}(A)\) then \(\Pi w = w\), and if \(v \notin \text{im}(A)\) then \(\Pi v \in \text{im}(A)\).

(Note that \(\Pi\) is only used in the analysis, and we don't need to actually construct it.)

There is a standard way to do it - by mapping each vector \(v \in \mathbb{R}^n\) into the vector \(w \in \text{im}(A)\) with \(\|v-w\|_2\) minimized, just like what we did in week 10 for the GMRES method.

For \(w \in \text{im}(A)\) to minimize \(\|v-w\|_2\), we must have \((v-w) \perp \text{im}(A)\),

which is equivalent to \(A^T(v-w) = 0\) \(\iff A^T v = A^T w = 0\).
Since \( w \in \text{Im}(A) \), it can be written as \( w = Ax \), and the above is equivalent to \( A^T v = A^T Ax \), and thus \( x = (A^T A)^{-1} A^T v \), and hence \( w = Ax = A(A^T)^{-1} A^T v \).

So \( \Pi = A(A^T)^{-1} A^T \) is the projection matrix.

In our case \( A = W^\perp B \), which is not a full rank matrix, but we can use the pseudo-inverse in place of the inverse \( (A^T A)^{-1} \).

Therefore, \( \Pi = W^\perp B (W^\perp B)^T \), which is \( \Pi = W^\perp B (B^T W B)^T W^\perp = W^\perp B (B^T W B)^T W^\perp \).

Properties of \( \Pi \)

1. \( \Pi^2 = \Pi \), true for any projection matrix.
2. \( \Pi \) has rank \( n-1 \), with 1 as an eigenvalue with multiplicity \( n-1 \).
3. \( \Pi \) is symmetric, and so \( \Pi e e^T = \Pi e e^T \) where \( e \) is the \( e \)-th column of \( \Pi \).
4. Let \( x = (v, v) \), then \( B L^{-1} B^T e e^T = (x - x) L^{-1} (x - x) = Re \), and so \( \Pi (x) = \Pi_{e e} \).

Matrix Concentration Result

Recall that we have reduced the problem to bound \( \| \Pi^T S_{n-1} - \Pi^T \Pi \|_2 \). Since \( \Pi \) is symmetric, this is equal to \( \| \Pi S_{n-1} - \Pi \Pi \|_2 \), which is equal to \( \| \sum \mathbb{E} (x, x) \Pi e e^T - \Pi \Pi \|_2 \).

This is some sort of concentration result for rank-1 matrices.

And you find out that it has been studied before. We simplify it slightly for our application.

**Theorem** (Rudelson and Vershynin [5])

Let \( y_1, ..., y_m \) be \( n \)-dimensional vectors with \( \| y_i \|_2 \leq M \) for all \( i \). Let \( p_1, ..., p_m \) be a probability distribution, with probability \( p_i \) of choosing \( y_i \).

Suppose \( \| \mathbb{E} [y y^T] \|_2 \leq 1 \). Let \( y_1, ..., y_m \) be independent samples drawn from \( p \).

Then \( \mathbb{E} \left\| \frac{1}{M} y y^T - \mathbb{E} [y y^T] \right\|_2 \leq O \left( M \sqrt{\frac{\log M}{M}} \right) \).

Let's apply this theorem and finish our proof.

Let \( y_i = \frac{1}{\sqrt{p_e}} M e \) with probability \( p_e \).

Then \( \frac{1}{M} y y^T = \frac{1}{M} e e^T \) (\( e \) of times \( e \) is sampled) \( \frac{\Pi e}{\sqrt{p_e}} \frac{\Pi^T e}{\sqrt{p_e}} = \sum \mathbb{E} (x, x) \Pi_{e e} \Pi_{e e} = \Pi S \Pi \).

Also, \( \mathbb{E} [y y^T] = \sum \mathbb{E} [y y^T] = M e e^T = \Pi \Pi \), and \( \| M e e^T \| = \| \Pi \Pi \| = \| \Pi \| = 1 \).
Therefore, the theorem tells us that $E \| \mathbf{WSU} - \mathbf{W} \|_2 \leq O\left(M \sqrt{\frac{\log \delta}{\delta}}\right)$.

Now we just need to compute $M$.

Note that $\| y \|_2 = \frac{1}{\sqrt{\delta}} \left\| \sum_{i=1}^{\delta} \mathbf{W} \right\|_2 = \frac{1}{\sqrt{\delta}} \sqrt{\sum_{i=1}^{\delta} \mathbf{w}_i^T \mathbf{w}_i}$.

by property \( \delta \) by property \( \delta \)

Then it is natural to set $p_0 = \frac{\mathbf{w}_0}{\mathbf{w}_0}$, and hence $\| y \|_2 = \sqrt{\sum_{i=1}^{\delta} \mathbf{w}_i^T \mathbf{w}_i}$.

Observe that $\sum_{i=1}^{\delta} \mathbf{w}_i = \sum_{i=1}^{\delta} \mathbf{W}_{i,e} = \text{Tr}(\mathbf{W}) = \text{Tr}(\mathbf{W}) = \text{sum of eigenvalues of } \mathbf{W} = \mathbf{W}$ by property $\delta$.

Therefore, $M = \mathbf{W}$.

So, by the theorem, we have $E \| \mathbf{WSU} - \mathbf{W} \|_2 \leq O\left(\sqrt{\frac{\log \delta}{\delta}}\right)$.

By setting $\delta = O\left(\log \log n / \epsilon^2\right)$, we have $E \| \mathbf{WSU} - \mathbf{W} \|_2 \leq \epsilon/2$.

Thus, with probability at least $1/2$, $\| \mathbf{WSU} - \mathbf{W} \|_2 \leq \epsilon$, and by the first main step this is equivalent to showing $(1-\epsilon)\mathbf{x}^T Q \mathbf{x} \leq \mathbf{x}^T \mathbf{L} \mathbf{x} \leq (1+\epsilon)\mathbf{x}^T Q \mathbf{x}$. We are done.

---

**Remarks**

- This is a randomized algorithm, with some error probability, and we can make it to be inverse polynomial of $n$, using a stronger version of the theorem that we didn't state.

- To construct the sparsifier, we need to compute $R_e$ for all $e$. A straightforward algorithm is to compute $Q_e^T$ but it is too slow. It turns out that a good approximation of $R_e$ is enough, and all approximate $R_e$ can be computed using $O(\log n)$ Laplacian solvers. Since Laplacian solvers can be implemented in $O(m)$ time, the algorithm can be done in $O(m)$ time. See [4] for details.

- Do you realize something strange? We said that the spectral sparsifier is used to design the near-linear time Laplacian solver, but we also just said that we need to use the Laplacian solver for the spectral sparsifier. This is circular! Don't worry. We will explain a nice way out of it.

- There is a truly beautiful algorithm of constructing a spectral sparsifier with only $O(n/\epsilon^2)$ edges. It was thought to be impossible even for cut sparsifiers. I see it as
a great triumph of the spectral technique for a seemingly purely combinatorial problem. See [6].

This can be seen as a (weighted) generalization of expander graphs, as the algorithm is deterministic.

But the construction algorithm is not fast enough (taking $O(n^6)$ time). Can you improve it?

References


