CPSC 536C - UBC WT2 2025-2026

Lecture : Spectral Sparsification

March 2, 2026

Lecturer: Tong Ling *Scribe: Vasiliki Mavrogeorgou*

1 Introduction

Many graph algorithms become inefficient when the graph is dense (ex. Bellman-Ford, Prim, Kruskal etc). A natural question is whether we can approximate a dense graph by one that has significantly fewer edges while preserving essential properties. Spectral sparsification provides a powerful framework for doing exactly this.

Our goal: Given a (dense) graph G (where $|E| = \omega(|V|^{1+\gamma})$, $\gamma \in (0, 1)$), we aim to construct a sparse graph H with significantly fewer edges (namely $|E| \in O(|V| \log^k(|V|))$, for some $k \in \mathbb{N}$) such that H approximates G in a strong spectral sense. Informally, this means that H preserves the overall structure of G and behaves in a similar way when we perform computations on it, despite having far fewer edges.

To achieve this, we represent graphs using their Laplacian matrices and compare graphs through these matrices. This allows us to use tools from linear algebra to analyze graph structure.

Structure of the lecture notes: We will study several approaches to constructing spectral sparsifiers, each offering a different way to understand the problem.

1. **Combinatorial approach:** We begin with Ramanujan graphs, which provide examples showing that very sparse graphs can still closely approximate dense ones.
2. **BSS framework:** We then introduce a general method for constructing sparsifiers for arbitrary graphs, achieving optimal sparsity.
3. **Optimization-based approach (Matrix Multiplicative Weights):** This viewpoint interprets sparsification as an iterative optimization process, giving a different way to think about the construction.
4. **Sampling-based approach:** Finally, we describe a simple algorithm based on sampling edges according to their importance, which leads to efficient implementations.

1.1 Graphs and Laplacians

1.1.1 Definition and Example

Let $G = (V, E, w)$ be a weighted undirected graph[1] with nonnegative edge weights.

Definition 1.1 (Graph Laplacian)**.** The Laplacian matrix $L_G \in \mathbb{R}^{n \times n}$ is defined by

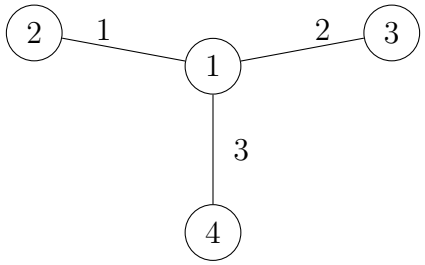
$$L_G(u, v) = \begin{cases} -w(u, v) & \text{if } u \neq v \text{ and } (u, v) \in E, \\ 0 & \text{if } u \neq v \text{ and } (u, v) \notin E, \\ \sum_z w(u, z) & \text{if } u = v. \end{cases}$$

✍ **Example:** For the following Laplacian

$$L_G = \begin{bmatrix} 6 & -1 & -2 & -3 \\ -1 & 1 & 0 & 0 \\ -2 & 0 & 2 & 0 \\ -3 & 0 & 0 & 3 \end{bmatrix}$$

,

we draw the star graph on 4 vertices with edge weights 1, 2 and 3, respectively:



The Laplacian captures how each vertex is connected to its neighbors. It plays a central role in many graph algorithms, including clustering, random walks, and electrical flows.

1.1.2 Why Laplacians?

The Laplacian provides a convenient way to encode graph structure. In particular, as we will encounter in the next section, expressions of the form $x^\top L_G x$, called *quadratic forms*, capture the total squared differences $(x(u) - x(v))^2$ across edges, meaning this quantity is large when the values of x change significantly across edges.

Since L_G is symmetric, graph G is completely determined by its quadratic form $x^\top L_G x$. This means that instead of comparing matrices entry-wise, we can compare how they act on vectors. This perspective allows us to compare graphs using matrix inequalities and leads to the notion of spectral sparsification.

[1] Throughout this lecture, we assume that G is connected for purposes of simplicity.

2 Spectral Sparsifiers

In this section, we introduce spectral sparsification through quadratic forms of graph Laplacians. We relate this notion to eigenvalues, give illustrative examples, and conclude with general existence results.

2.1 Graph Laplacians and Quadratic Forms

Let $G = (V, E, w)$ be a weighted undirected graph with nonnegative edge weights and $L_G \in \mathbb{R}^{n \times n}$ its Laplacian (view def. 1.1). Below, we will discuss some useful definitions and properties:

Definition 2.1 (Positive semidefinite matrix)**.** A symmetric matrix $M \in \mathbb{R}^{n \times n}$ is *positive semidefinite (PSD)* iff $\forall x \in \mathbb{R}^n$

$$x^\top M x \geq 0$$

Definition 2.2 (Löwner order)**.** For symmetric matrices A, B, we write $A \preceq B$ if $B - A$ is PSD.

Given positive semidefiniteness is a key property of our Laplacians, we need to derive some equivalent formulations to facilitate our analysis. Below, we present equivalent formulations of Positive-semidefiniteness:

Proposition 2.3. *Let M be a symmetric matrix. Then the following are equivalent:*

1. *M is positive-semidefinite*
2. *all of its eigenvalues are non-negative*
3. *there exists a matrix L s.t. $M = LL^\top$.*

Proof. (1) $\Rightarrow$ (2): Let v be an eigenvector with eigenvalue λ. Then

$$v^\top M v = \lambda \|v\|^2 \geq 0,$$

so $\lambda \geq 0$.

(2) $\Rightarrow$ (3): Let $M = U \Lambda U^\top$ be an eigendecomposition with Λ diagonal and nonnegative. Then we can write

$$M = U \Lambda^{1/2} \Lambda^{1/2} U^\top = (U \Lambda^{1/2})(U \Lambda^{1/2})^\top.$$

(3) $\Rightarrow$ (1): If $M = LL^\top$, then

$$x^\top M x = x^\top L L^\top x = \|L^\top x\|^2 \geq 0.$$

□

In our introduction, we had mentioned that Laplacians are particularly interesting, since they can give us information about the value differences across edges of a graph. In this Lemma, we prove that the Laplacian L_G of an undirected graph G admits a particularly useful quadratic form that allows us to decode this structure:

Lemma 2.4. *For any vector $x \in \mathbb{R}^n$,*

$$x^\top L_G x = \sum_{(u,v)\in E} w(u,v)(x(u) - x(v))^2.$$

Proof. Recall that $(L_G x)(u) = \sum_v w(u,v)(x(u) - x(v))$. Then

$$x^\top L_G x = \sum_u x(u)(L_G x)(u) = \sum_{u,v} w(u,v)x(u)(x(u) - x(v)).$$

By symmetry (swapping u, v and averaging), this becomes

$$\frac{1}{2}\sum_{u,v} w(u,v)(x(u) - x(v))^2,$$

which equals the desired expression since the graph is undirected. □

✎ **Insight:** Something worth mentioning from the quadratic form expression above is that

$$x^\top L_G x = \sum_{(u,v)\in E} w(u,v)(x(u) - x(v))^2 \geq 0,$$

so L_G is **positive semidefinite**. Moreover, each row of L_G sums to zero, which implies that $L_G\mathbf{1} = 0$ so 0 is an **eigenvalue**. In the next lemma, we will prove that for a connected graph, the multiplicity of its 0 eigenvalue is one.

Lemma 2.5. *If G is connected, then L_G has exactly one zero eigenvalue.*

Proof. The nullspace of L_G consists of vectors x such that $x(u) = x(v)$ for every edge (u, v). Thus, x must be constant on each connected component.

If G is connected, this implies that x is constant on all vertices, so the nullspace is one-dimensional. □

2.2 Spectral Approximation

We now define the **central** notion of this section.

Definition 2.6 (Spectral sparsifier)**.** Let G and H be graphs on the same vertex set, with H sparse (significantly fewer edges than G). We say that H is a $(1+\epsilon)$-**spectral sparsifier** of G if for all $x \in \mathbb{R}^n$,

$$x^\top L_G x \leq x^\top L_H x \leq (1+\epsilon) x^\top L_G x.$$

Equivalently,

$$L_G \preceq L_H \preceq (1+\epsilon) L_G.$$

This definition means that H preserves the quantities $x^\top L_G x$ up to a small error, so it behaves similarly to G when we measure how values change across the graph.

Lemma 2.7. *Assume the eigenvalues are ordered decreasingly. If*

$$L_G \preceq L_H \preceq (1+\epsilon) L_G,$$

then for all i,

$$\lambda_i(L_G) \leq \lambda_i(L_H) \leq (1+\epsilon)\lambda_i(L_G).$$

Proof. Our eigenvalues are ordered decreasingly, thus we use the max-min principle of eigenvalues:

$$\lambda_i(A) = \max_{\dim S = i} \min_{x \in S,\, x \neq 0} \frac{x^\top A x}{x^\top x}.$$

which allows us to approximate eigenvalues without fully diagonalizing. Since

$$x^\top L_G x \leq x^\top L_H x \leq (1+\epsilon) x^\top L_G x$$

for all x, the same inequalities hold for what is known as the Rayleigh quotient:

$$\frac{x^\top L_G x}{x^\top x} \leq \frac{x^\top L_H x}{x^\top x} \leq (1+\epsilon)\frac{x^\top L_G x}{x^\top x}.$$

Taking the minimum over $x \in S$ and then the maximum over all subspaces S of a specific dimension i, preserves the inequalities, thus yielding the result. □

2.3 Example: Complete Graph and Ramanujan Graphs

We now illustrate spectral sparsification through an example.

Consider the complete graph G on n vertices. Its Laplacian is:

$$L_G = \begin{pmatrix} n-1 & -1 & \cdots & -1 \\ -1 & n-1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & n-1 \end{pmatrix}.$$

It is a useful exercise to verify that the eigenvalues of L_G are

$$\lambda_1 = \cdots = \lambda_{n-1} = n, \quad \lambda_n = 0.$$

To understand how well a sparse graph can approximate a dense one, we look for graphs whose eigenvalues are as tightly controlled as possible. Thus, we introduce **Ramanujan** graphs, which are known to achieve optimal spectral bounds for sparse regular graphs, making them a natural candidate for spectral sparsification. Recall that a d-regular graph is a graph where every vertex has the exact same degree d.

Definition 2.8. A d-regular graph H is called *Ramanujan* if every nontrivial eigenvalue μ of its adjacency matrix satisfies

$$|\mu| \leq 2\sqrt{d-1}.$$

Since H is d-regular, its Laplacian is $L_H = dI - A_H$. Therefore, the nonzero Laplacian eigenvalues of H lie in the interval

$$d - 2\sqrt{d-1} \leq \lambda_i(L_H) \leq d + 2\sqrt{d-1}.$$

Claim 2.9. *A suitably rescaled d-regular Ramanujan graph H is a spectral sparsifier of the complete graph K_n.*

Proof of Claim. Let $G = K_n$. The Laplacian of G satisfies

$$x^\top L_G x = n\|x\|^2$$

for every vector $x \perp \mathbf{1}$. In particular, if $\|x\| = 1$, then

$$x^\top L_G x = n.$$

Now let H be a d-regular Ramanujan graph. Since H is connected, the only zero eigenvalue of L_H corresponds to the vector $\mathbf{1}$. Hence it suffices to compare quadratic forms on the subspace $\mathbf{1}^\perp$.

For every unit vector $x \perp \mathbf{1}$, the Rayleigh quotient of L_H satisfies

$$d - 2\sqrt{d-1} \leq x^\top L_H x \leq d + 2\sqrt{d-1}.$$

Using $x^\top L_G x = n$ on this same subspace, we get

$$\frac{d - 2\sqrt{d-1}}{n} x^\top L_G x \leq x^\top L_H x \leq \frac{d + 2\sqrt{d-1}}{n} x^\top L_G x.$$

Now rescale every edge of H by

$$\frac{n}{d - 2\sqrt{d-1}}.$$

Equivalently, the new Laplacian is

$$\widetilde{L}_H = \frac{n}{d - 2\sqrt{d-1}} L_H.$$

Multiplying the previous inequality by this factor gives

$$x^\top L_G x \le x^\top \widetilde{L}_H x \le \frac{d+2\sqrt{d-1}}{d-2\sqrt{d-1}} x^\top L_G x$$

for all $x \perp \mathbf{1}$.

Both L_G and $\widetilde{L}_H$ vanish on the span of $\mathbf{1}$, so the same inequality holds for all vectors $x \in \mathbb{R}^n$. Thus the rescaled Ramanujan graph is a $\frac{d+2\sqrt{d-1}}{d-2\sqrt{d-1}}$-spectral approximation of K_n. Since H has only $nd/2$ edges, this gives a sparse spectral approximation of the complete graph. □

The key idea behind this is that the complete graph has all nonzero Laplacian eigenvalues equal to n. Ramanujan graphs have all nonzero Laplacian eigenvalues tightly concentrated around d. After rescaling, this makes a Ramanujan graph a good spectral sparsifier of K_n.

The BSS theorem extends this phenomenon beyond the complete graph: it shows that every weighted graph admits a sparse spectral approximation. One way to state the result is the following.

Theorem 2.10 (BSS / Twice-Ramanujan sparsifiers [2])**.** *For every $d > 1$ and every undirected, weighted graph $G = (V, E, w_G)$ on n vertices, there exists a weighted graph $H = (V, F, w_H)$ with $|F| \le \lceil d(n-1) \rceil$ edges such that*

$$L_G \preceq L_H \preceq \left(\frac{d+1+2\sqrt{d}}{d+1-2\sqrt{d}} \right) L_G.$$

The term *twice-Ramanujan* comes from comparing this theorem with the Ramanujan example above. A d-regular Ramanujan graph has $dn/2$ edges, while the BSS sparsifier has at most $\lceil d(n-1) \rceil$ edges. Thus, BSS achieves a comparable spectral approximation for arbitrary graphs using roughly twice as many edges as the Ramanujan graph uses for the complete graph.

2.3.1 Reduction to a linear-algebraic problem

To prove Theorem 2.10, we state an equivalent formulation of the sparsification problem in terms of decomposing the identity matrix into rank-one matrices.

Theorem 2.11. *Suppose $d > 1$. Let $v_1, \dots, v_m \in \mathbb{R}^n$ satisfy $\sum_{i=1}^m v_i v_i^\top = I_n$. Then there exist scalars $s_i \ge 0$ with $|\{i : s_i \ne 0\}| \le dn$ such that*

$$I_n \preceq \sum_{i=1}^m s_i v_i v_i^\top \preceq \left(\frac{d+1+2\sqrt{d}}{d+1-2\sqrt{d}} \right) I_n.$$

Why Theorem 2.11 implies Theorem 2.10.

Proof. Let $\alpha_d := \frac{d+1+2\sqrt{d}}{d+1-2\sqrt{d}}$ and $G = (V, E, w)$ be a weighted graph with Laplacian L_G. Recall that

$$L_G = \sum_{e=(u,v)\in E} w_e(e_u - e_v)(e_u - e_v)^\top.$$

Indeed, for any vector $x \in \mathbb{R}^n$,

$$x^\top L_G x = \sum_{e=(u,v)\in E} w_e(x_u - x_v)^2 = x^\top \left(\sum_{e=(u,v)\in E} w_e(e_u - e_v)(e_u - e_v)^\top \right) x.$$

Since $L_G\mathbf{1} = 0$, the Laplacian is singular (eigenvalue 0). We therefore work on the subspace $\mathbf{1}^\perp = \{x \in \mathbb{R}^n : x^\top \mathbf{1} = 0\}$, where L_G is invertible.

For each edge $e = (u, v)$, we define $b_e := e_u - e_v$, and set $v_e := \sqrt{w_e}\,(L_G^+)^{1/2} b_e$, where L_G^+ denotes the pseudoinverse of L_G.

Then, we get that:

$$\sum_{e\in E} v_e v_e^\top = \sum_{e\in E} w_e (L_G^+)^{1/2} b_e b_e^\top (L_G^+)^{1/2}.$$

Using the decomposition of the Laplacian $\sum_{e\in E} w_e b_e b_e^\top = L_G$, we obtain

$$\sum_{e\in E} v_e v_e^\top = (L_G^+)^{1/2} L_G (L_G^+)^{1/2}.$$

The matrix $(L_G^+)^{1/2} L_G (L_G^+)^{1/2}$ acts as the identity on $\mathbf{1}^\perp$. Thus, the vectors $\{v_e\}_{e\in E}$ form a decomposition of the identity on this subspace.

Applying Theorem 2.11, we obtain coefficients $s_e \geq 0$, with at most dn nonzero coefficients, such that

$$I \preceq \sum_{e\in E} s_e v_e v_e^\top \preceq \alpha_d I$$

on $\mathbf{1}^\perp$. Substituting the definition of v_e, this becomes

$$I \preceq (L_G^+)^{1/2} \left(\sum_{e\in E} s_e w_e b_e b_e^\top \right) (L_G^+)^{1/2} \preceq \alpha_d I.$$

We then define a graph H on the same vertex set by assigning edge e weight $w_H(e) := s_e w_e$. Then $L_H = \sum_{e\in E} s_e w_e b_e b_e^\top$. Hence, $I \preceq (L_G^+)^{1/2} L_H (L_G^+)^{1/2} \preceq \alpha_d I$.

Multiplying on the left and right by $L_G^{1/2}$ on $\mathbf{1}^\perp$ gives

$$L_G \preceq L_H \preceq \frac{d+1+2\sqrt{d}}{d+1-2\sqrt{d}} L_G.$$

Since both L_G and L_H vanish on the span of $\mathbf{1}$, the inequality holds on all of $\mathbb{R}^n$.

Finally, since at most dn coefficients s_e are nonzero, the graph H contains at most dn edges. Therefore, H is a spectral sparsifier of G satisfying the conclusion of Theorem 2.10. □

The following result extends the BSS theorem from rank-one matrices to general positive semidefinite matrices, and is stated and proved by de Carli Silva, Harvey, and Sato [3]:

Theorem 2.12. *Let $B_1, \dots, B_m$ be symmetric (or Hermitian), positive semidefinite $n \times n$ matrices of arbitrary rank and $B := \sum_i B_i$. Then, for any $\varepsilon \in (0, 1)$, there exist nonnegative $\mathbf{s} = (s_1, \dots, s_m) \in \mathbb{R}^m$, with $O(n/\varepsilon^2)$ nonzero entries, such that*

$$B \preceq \sum_i s_i B_i \preceq (1+\varepsilon)B.$$

The algorithm computing $s_1, \dots, s_m$ runs in $O(mn^3/\varepsilon^2)$ time.

✎ **Insight:** We highlight that in the BSS theorem, the matrices are of rank 1, but De Carli Silva's work generalizes the BSS method of finding the coefficients to PSD matrices.

Now, we collect here several tools that will be used in the analysis of the BSS construction.

- **Eigenvalue interlacing:** Let A be PSD with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_{n-1} > \lambda_n$, and let $A + vv^T$, $v \neq 0$, have eigenvalues $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_{n-1} > \mu_n$. Then the eigenvalues interlace:
$$\lambda_{n-1} \leq \mu_{n-1} \leq \lambda_{n-2} \leq \cdots \leq \lambda_1 < \mu_1.$$
This describes how a rank-one update affects the spectrum.

- **Sherman-Morrison formula:** If A is a nonsingular $n \times n$ matrix and v is a vector, then
$$(A + vv^T)^{-1} = A^{-1} - \frac{A^{-1}vv^TA^{-1}}{1 + v^TA^{-1}v}.$$
This allows us to update inverses efficiently under rank-one updates.

- **Trace properties:** The trace is linear and satisfies $\mathrm{Tr}(XY) = \mathrm{Tr}(YX)$. *These properties will be used repeatedly when manipulating potential functions.*

Definition 2.13 (potential functions)**.** Let A be a symmetric matrix with eigenvalues $\lambda_1, \dots, \lambda_n$. Define
$$\Phi^u(A) := \mathrm{Tr}(uI - A)^{-1} = \sum_i \frac{1}{u - \lambda_i},$$
and
$$\Phi_l(A) := \mathrm{Tr}(A - lI)^{-1} = \sum_i \frac{1}{\lambda_i - l}.$$
We require $\lambda_{\min} > l$ and $\lambda_{\max} < u$. They measure how far away the eigenvalues of A are from u and l.

Lemma 2.14. *This algorithm constructs a sequence of matrices $0 = A^{(0)}, A^{(1)}, \ldots, A^{(dn)}$, obtaining $A^{(j+1)}$ by adding tvv^T (for some $t \geq 0$ and $v \in \{v_i\}$) to $A^{(j)}$. The goal is for $A^{(dn)}$ to satisfy*

$$\frac{\lambda_{\max}(A^{(dn)})}{\lambda_{\min}(A^{(dn)})} \leq \frac{u_0 + dn\delta_U}{l_0 + dn\delta_L} = \frac{d+1+2\sqrt{d}}{d+1-2\sqrt{d}}.$$

Sketch of proof.

$$\Phi^{u'}(A + tvv^T) = \mathrm{Tr}(u'I - A - tvv^T)^{-1} = \Phi^{u'}(A) + \frac{v^T(u'I - A)^{-2}v}{\frac{1}{t} - v^T(u'I - A)^{-1}v}.$$

So,

$$\Phi^{u'}(A+tvv^T) \leq \Phi^{u}(A) \iff \frac{1}{t} \geq v^T(u'I-A)^{-1}v + \frac{v^T(u'I - A)^{-2}v}{\Phi^{u}(A) - \Phi^{u'}(A)} = v^T(U_A)v = \mathrm{Tr}(U_A vv^T).$$

This is linear in vv^T.
Similarly, for the lower barrier. Let $l' = l + \delta_L$. We have

$$\Phi_{l'}(A + tvv^T) = \Phi_{l'}(A) - \frac{v^T(A - l'I)^{-2}v}{\frac{1}{t} + v^T(A - l'I)^{-1}v}.$$

So,

$$\Phi_{l'}(A + tvv^T) \leq \Phi_{l}(A) \iff \frac{1}{t} \leq v^T \left(\frac{(A - l'I)^{-2}}{\Phi_{l'}(A) - \Phi_l(A)} - (A - l'I)^{-1} \right) v = \mathrm{Tr}(L_A vv^T).$$

Now we prove that there exists v such that

$$\mathrm{Tr}(U_A vv^T) \leq \mathrm{Tr}(L_A vv^T),$$

because if so, there exists t such that

$$\mathrm{Tr}(U_A vv^T) \leq \frac{1}{t} \leq \mathrm{Tr}(L_A vv^T).$$

We have

$$\sum_i \mathrm{Tr}(U_A v_i v_i^T) = \mathrm{Tr}\left(U_A \left(\sum_i v_i v_i^T \right) \right) = \mathrm{Tr}(U_A I_n) = \mathrm{Tr}(U_A).$$

If $\mathrm{Tr}(U_A) \leq \mathrm{Tr}(L_A)$, then such a vector v exists; otherwise a contradiction.
We have

$$\mathrm{Tr}(U_A) = \frac{\mathrm{Tr}(u'I - A)^{-2}}{\Phi^{u}(A) - \Phi^{u'}(A)} + \mathrm{Tr}(u'I - A)^{-1}.$$

The last term equals $\Phi^{u'}(A) \leq \Phi^{u}(A) \leq \varepsilon_U$. The numerator equals $-\frac{\partial}{\partial u'}\Phi^{u'}(A)$. The denominator $\geq \delta_U\big(-\frac{\partial}{\partial u'}\Phi^{u'}(A)\big)$ by convexity of the potential function. (convex function: $f(y) \geq f(x) +$

$f'(x)(y-x)$.) So $\mathrm{Tr}(U_A) \le \frac{1}{\delta_U} + \varepsilon_U$.
Similarly, we have $\mathrm{Tr}(L_A) \ge \frac{1}{\delta_L} - \varepsilon_L$. So it suffices to require

$$\frac{1}{\delta_U} + \varepsilon_U \le \frac{1}{\delta_L} - \varepsilon_L.$$

Parameters: $\delta_L = 1$, $\varepsilon_L = \frac{1}{\sqrt{d}}$, $l_0 = -\frac{n}{\varepsilon_L}$, $\delta_U = \frac{\sqrt{d}+1}{\sqrt{d}-1}$, $\varepsilon_U = \frac{\sqrt{d}-1}{d+\sqrt{d}}$, $u_0 = \frac{n}{\varepsilon_U}$. Then, after dn steps,

$$\frac{\lambda_{\max}(A^{(dn)})}{\lambda_{\min}(A^{(dn)})} \le \frac{d + 2\sqrt{d} + 1}{d - 2\sqrt{d} + 1},$$

as desired. □

3 MMWUM as Another Solution

The BSS construction shows that spectral sparsifiers exist, but its proof is combinatorial and technically involved. In this section, we present an alternative viewpoint based on optimization, interpreting sparsification as a regret minimization problem over matrices.

Our goal: We aim to construct a matrix $A = \sum_{i=1}^{m} s_i B_i$ that approximates the identity matrix in the sense that $I \preceq A \preceq (1+\epsilon)I$, while keeping only few nonzero coefficients s_i. This corresponds to selecting a small subset of edges in the graph.

3.1 Setup

Let $B_1, \ldots, B_m \in \mathbb{S}_+^n$ be PSD matrices such that $\sum_{i=1}^{m} B_i = I$. We iteratively construct

$$A(t) = \sum_{s=1}^{t} \alpha(s) B_{j(s)},$$

where at each step we choose an index $j(t)$ and a weight $\alpha(t) > 0$.

3.1.1 Potential Functions

To control the spectrum of $A(t)$, we define two matrix-valued potentials:

$$W_U(t) = \exp(\gamma A(t)), \quad W_L(t) = \exp(-\gamma A(t)),$$

and we track their traces

$$\Phi_U(t) = \mathrm{Tr}(W_U(t)), \qquad \Phi_L(t) = \mathrm{Tr}(W_L(t)).$$

We also define normalized matrices:

$$X_U(t) = \frac{W_U(t)}{\mathrm{Tr}(W_U(t))}, \quad X_L(t) = \frac{W_L(t)}{\mathrm{Tr}(W_L(t))}.$$

✎ **Insight:** Intuitively, Φ_U penalizes large eigenvalues of $A(t)$, whereas Φ_L penalizes small eigenvalues. Controlling both ensures that all eigenvalues remain in a desired range.

At the same time, this update rule resembles multiplicative weights: we maintain a distribution over directions and update it based on observed losses encoded by the matrices B_i.

3.1.2 Main Guarantee

The goal of the update procedure is to ensure that the average iterate remains spectrally close to the identity.

Theorem 3.1. *Suppose the updates $(\alpha(t), j(t))$ satisfy the conditions below. Then after $T = O(n \log n/\epsilon^2)$ iterations,*

$$I \preceq \frac{A(T)}{T} \preceq (1+\epsilon)I.$$

In particular, this yields a spectral sparsifier using only $O(n \log n/\epsilon^2)$ nonzero terms.

3.1.3 Update Rule

At each iteration, we choose $(\alpha(t), j(t))$ satisfying:

$$\delta_U \geq \frac{e^{\gamma\alpha(t)} - 1}{\alpha(t)} \langle X_U(t), B_{j(t)} \rangle, \tag{1}$$

$$\delta_L \leq \frac{1 - e^{-\gamma\alpha(t)}}{\alpha(t)} \langle X_L(t), B_{j(t)} \rangle. \tag{2}$$

These conditions ensure that the upper potential Φ_U does not increase too quickly, and the lower potential Φ_L does not decrease too quickly.

The following lemma is the key step in proving the theorem.

Lemma 3.2. *At each iteration,*

$$\Phi_U(t+1) \leq (1+\delta_U)\Phi_U(t), \quad \Phi_L(t+1) \leq (1-\delta_L)\Phi_L(t).$$

Proof sketch. The proof follows from matrix exponential inequalities and linearization of the trace. The update conditions are designed precisely to enforce these bounds. □

Iterating the bounds. Iterating the inequalities gives

$$\Phi_U(T) \le (1+\delta_U)^T \Phi_U(0), \quad \Phi_L(T) \le (1-\delta_L)^T \Phi_L(0).$$

Since $A(0) = 0$, we have

$$\Phi_U(0) = \Phi_L(0) = \mathrm{Tr}(I) = n.$$

Hence

$$\Phi_U(T) \le (1+\delta_U)^T n, \quad \Phi_L(T) \le (1-\delta_L)^T n.$$

3.2 Bounding Eigenvalues

We now relate the potentials to the extreme eigenvalues of $A(T)$ in order to prove the theorem.

Since

$$\mathrm{Tr}(\exp(\gamma A)) = \sum_i e^{\gamma \lambda_i(A)} \ge e^{\gamma \lambda_{\max}(A)},$$

we obtain

$$e^{\gamma \lambda_{\max}(A(T))} \le \Phi_U(T).$$

Taking logarithms gives

$$\lambda_{\max}(A(T)) \le \frac{T \log(1+\delta_U) + \log n}{\gamma}.$$

Similarly, using

$$\mathrm{Tr}(\exp(-\gamma A)) = \sum_i e^{-\gamma \lambda_i(A)} \ge e^{-\gamma \lambda_{\min}(A)},$$

we obtain

$$e^{-\gamma \lambda_{\min}(A(T))} \le \Phi_L(T),$$

which implies

$$\lambda_{\min}(A(T)) \ge \frac{T \log((1-\delta_L)^{-1}) - \log n}{\gamma}.$$

Combined bound. Combining both inequalities yields

$$\left(\frac{\log(1-\delta_L)^{-1}}{\gamma} - \frac{\log n}{T\gamma} \right) I \preceq \frac{A(T)}{T} \preceq \left(\frac{\log(1+\delta_U)}{\gamma} + \frac{\log n}{T\gamma} \right) I.$$

Thus, the average iterate $A(T)/T$ is spectrally close to the identity, proving the theorem.

Conclusion: By choosing parameters appropriately, we ensure

$$I \preceq \frac{A(T)}{T} \preceq (1+\epsilon)I,$$

while using only $T = O(n \log n/\epsilon^2)$ nonzero terms.

This yields a spectral sparsifier in the transformed space; mapping back to the Laplacian decomposition gives a sparsifier for the graph.

3.3 Comparison with BSS

We can reinterpret the potentials in a barrier-function form:

$$\Psi^u(A) := \mathrm{Tr}(\exp(-uI + \gamma A)), \quad \Psi_l(A) := \mathrm{Tr}(\exp(lI - \gamma A)).$$

At each step, updating $A(t)$ while controlling Φ_U and Φ_L is equivalent to ensuring that these barrier potentials do not increase.

✎ **Insight:** The key difference between BSS and MMWUM is that:

- original BSS uses barrier functions and rank-one updates,
- MMWUM uses mirror descent and exponential potentials.

Both the BSS framework and the MMWUM approach show that spectral sparsifiers exist and can be constructed by carefully controlling eigenvalues through iterative updates. However, these methods are primarily theoretical and do not immediately yield the most efficient algorithms in practice.

4 Effective Resistance Sampling

Algorithmic perspective. The previous sections established that spectral sparsifiers exist and can be constructed via iterative spectral control. We now present a more direct and algorithmic approach based on *effective resistances*, which leads to simple and efficient sparsification procedures.

4.1 Electrical Interpretation

We view the graph $G = (V, E, w)$ as an electrical network where each edge $e = (u, v)$ is a resistor with conductance w_e (i.e., resistance $1/w_e$).

Definition 4.1 (Effective resistance)**.** The effective resistance of an edge $e = (u, v)$ is defined as

$$R_e = (e_u - e_v)^\top L_G^+ (e_u - e_v),$$

where L_G^+ is the Moore-Penrose pseudoinverse of the Laplacian.

Since $L_G \mathbf{1} = 0$, the Laplacian is singular and therefore not invertible. For this reason, we work with the pseudoinverse L_G^+ instead.

The Laplacian pseudoinverse plays an important role in quantities such as effective resistance. Since the pseudoinverse is obtained by inverting the nonzero eigenvalues of the Laplacian, spectral approximation naturally extends to the pseudoinverse:

$$\frac{1}{1+\epsilon} x^\top L_G^+ x \;\le\; x^\top L_H^+ x \;\le\; \frac{1}{1-\epsilon} x^\top L_G^+ x \quad \text{for all } x \in \mathbb{R}^n.$$

Since effective resistance can be written as $R_{uv} = (e_u - e_v)^\top L_G^+ (e_u - e_v)$, it follows that spectral sparsifiers approximately preserve effective resistances. Additionally, since L_G is positive semidefinite, its *pseudoinverse* L_G^+ is also **positive semidefinite**. Thus $R_e \ge 0$.

✎ **Insight:** The effective resistance R_e can also be interpreted as the potential difference induced when one unit of current is injected at u and removed at v. It measures how important an edge is for connectivity:

- If R_e is large, the edge is crucial (few alternative paths),
- If R_e is small, the edge is redundant (many alternative paths).

4.2 Approximating Effective Resistances

Theorem 4.2 (Approximate Effective Resistances)**.** *There exists a nearly-linear time algorithm which, with probability at least* $1 - \frac{1}{n}$*, given* $\epsilon > 0$ *and a weighted graph* $G = (V, E, w)$*, computes a matrix* $Z \in \mathbb{R}^{k \times n}$*, where* $k = O\big(\frac{\log n}{\epsilon^2}\big)$ *, such that*

$$(1-\epsilon) R_{uv} \le \|Z(e_u - e_v)\|^2 \le (1+\epsilon) R_{uv}$$

for every pair of vertices $u, v \in V$*.*

The running time is $\tilde{O}\big(m \frac{\log r}{\epsilon^2}\big)$, where $r = \frac{w_{\max}}{w_{\min}}$ is the ratio between the largest and smallest edge weights in the graph.

Modern Laplacian solvers allow us to approximately compute $L_G^+ x$ in nearly-linear time. Using this, one can estimate effective resistances via expressions of the form $R_e = (e_u - e_v)^\top L_G^+ (e_u - e_v)$.

Computing these quantities for all edges directly would be expensive. Instead, we use Johnson–Lindenstrauss as follows: we compute projected vectors corresponding to $L_G^+ e_u$ for each vertex u, using a small number of Laplacian linear system solves. Then, for an edge $e = (u, v)$, we estimate

$$R_e = (e_u - e_v)^\top L_G^+ (e_u - e_v)$$

by taking the squared distance between the corresponding projected vectors for u and v. Since Johnson–Lindenstrauss approximately preserves squared distances, these estimates are accurate for all edges simultaneously. Once the projected vertex vectors are computed, estimating all edge resistances only requires one pass over the edge set, which takes $O(m)$ time up to logarithmic factors. Thus, this avoids computing each effective resistance separately and leads to a nearly-linear time algorithm.

4.3 Sampling Scheme

We construct a sparsifier H by sampling edges independently. For each edge e, we sample with probability

$$p_e = \frac{w_e R_e}{\sum_{f \in E} w_f R_f}.$$

We sample q edges independently according to this distribution. Each sampled edge e is added to H with weight:

$$\widetilde{w}_e = \frac{w_e}{q p_e}.$$

This sampling distribution is **crucial** because edges with large effective resistance are more critical for connectivity, so they are sampled more often and lastly, it ensures that each sampled edge gives an unbiased estimate of its contribution to the Laplacian, so that $\mathbb{E}[L_H] = L_G$.

4.4 Sparsify Algorithm

Sparsify(G, q)
Input: Graph $G = (V, E, w)$, number of samples q
Procedure:

1. Initialize H as an empty graph.
2. For $i = 1, \ldots, q$:
   - Sample edge e with probability $p_e \propto w_e R_e$,
   - Add e to H with weight $\widetilde{w}_e = \frac{w_e}{q p_e}$.
3. If an edge is sampled multiple times, sum its weights.

Output: Sparsified graph H

By sampling edges according to effective resistance, we obtain a sparse graph with $O(n \log n/\epsilon^2)$ edges that spectrally approximates the original graph.

Lemma 4.3. *Let L_H be the Laplacian of the sampled graph. Then*

$$\mathbb{E}[L_H] = L_G.$$

Proof. Each sampled edge $e = (u, v)$ contributes the matrix

$$\frac{w_e}{q p_e}(e_u - e_v)(e_u - e_v)^\top.$$

Let X_i be the contribution of the i-th sampled edge. Then

$$L_H = \sum_{i=1}^{q} X_i \Rightarrow \mathbb{E}[L_H] = \sum_{i=1}^{q} \mathbb{E}[X_i],$$

after taking expectation. Since each edge is sampled with probability p_e,

$$\mathbb{E}[X_i] = \sum_{e \in E} p_e \cdot \frac{w_e}{q p_e}(e_u - e_v)(e_u - e_v)^\top = \frac{1}{q} \sum_{e \in E} w_e (e_u - e_v)(e_u - e_v)^\top.$$

Summing over $i = 1, \ldots, q$ gives

$$\mathbb{E}[L_H] = L_G.$$

□

Lemma 4.4. *If G is connected, then $\sum_{e \in E} w_e R_e = n - 1$.*

Proof. Recall that the effective resistance of an edge $e = (u, v)$ is

$$R_e = (e_u - e_v)^\top L_G^+ (e_u - e_v).$$

Using the identity $x^\top A x = \mathrm{Tr}(Axx^\top)$, we can rewrite this as

$$R_e = \mathrm{Tr}\big(L_G^+ (e_u - e_v)(e_u - e_v)^\top\big).$$

Summing over all edges and multiplying by w_e, we obtain

$$\sum_{e \in E} w_e R_e = \sum_{e \in E} w_e \,\mathrm{Tr}\big(L_G^+ (e_u - e_v)(e_u - e_v)^\top\big) = \mathrm{Tr}\left(L_G^+ \sum_{e=(u,v)\in E} w_e (e_u - e_v)(e_u - e_v)^\top \right),$$

where the last equality occurs from the linearity of trace. The sum inside is exactly the Laplacian L_G, so

$$\sum_{e \in E} w_e R_e = \mathrm{Tr}(L_G^+ L_G).$$

Finally, since G is connected, the Laplacian L_G has eigenvalues $\lambda_1 \geq \cdots \geq \lambda_{n-1} > 0$ and $\lambda_n = 0$. Thus L_G^+ has eigenvalues $1/\lambda_1, \ldots, 1/\lambda_{n-1}, 0$, and $L_G L_G^+$ has eigenvalues $1, \ldots, 1, 0$. Therefore,

$$\mathrm{Tr}(L_G L_G^+) = n - 1,$$

which proves the claim. □

✎ **Insight:** If $e = (u, v)$ is a bridge, then $R_e = \frac{1}{w_e}$.

The previous lemmas show that the sampling procedure is unbiased and well-scaled. We conclude our notes by showing that with sufficiently many samples, it yields a spectral sparsifier.

We will use the following matrix concentration result, which is the one used by Spielman and Srivastava in the proof of their sparsification theorem.

Lemma 4.5 (Rudelson-Vershynin, informal)**.** *Let p be a probability distribution over $\Omega \subseteq \mathbb{R}^d$ such that $\sup_{y \in \Omega} \|y\|_2 \leq M$ and $\|\mathbb{E}_p y y^T\|_2 \leq 1$. Let $y_1 \ldots y_q$ be independent samples drawn from p. Then*

$$\mathbb{E} \left\| \frac{1}{q} \sum_{i=1}^{q} y_i y_i^T - \mathbb{E} y y^T \right\|_2 \leq \min\left(CM \sqrt{\frac{\log q}{q}}, 1 \right)$$

where C is an absolute constant.

Theorem 4.6. *Let $q = O\big(\frac{n \log n}{\epsilon^2}\big)$. Then with probability at least $1/2$,*

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

Proof sketch. We omit the full reduction used in the paper, which rewrites the sampling process in terms of a projection matrix, and focus on the role of the concentration lemma.

We write L_H as a sum of independent random PSD matrices. The analysis relies on:

- unbiasedness $\mathbb{E}[L_H] = L_G$,
- bounding the size of the sampled matrices using effective resistances,
- the Rudelson-Vershynin matrix concentration lemma for sums of random rank-one matrices.

To use the lemma above, we rewrite the sampling process as sampling random rank-one matrices of the form $yy^\top$. Effective-resistance sampling ensures that these matrices have controlled norm, while the unbiasedness calculation gives the correct expectation. Applying the Rudelson-Vershynin lemma then shows that the sampled sum concentrates around its expectation once $q = O\big(\frac{n \log n}{\epsilon^2}\big)$, implying that

$$(1 - \epsilon)L_G \preceq L_H \preceq (1 + \epsilon)L_G,$$

which concludes the proof. □

✎ **Insight:** The probability at least $1/2$ in the theorem statement comes from applying Markov's inequality to the expectation bound obtained from the Rudelson-Vershynin lemma.

References

[1] Zeyuan Allen-Zhu, Zhenyu Liao, and Lorenzo Orecchia. Spectral sparsification and regret minimization beyond matrix multiplicative updates, 2015.

[2] Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-Ramanujan Sparsifiers. *CoRR*, abs/0808.0163, 2008.

[3] Marcel Kenji de Carli Silva, Nicholas J. A. Harvey, and Cristiane M. Sato. Sparse sums of positive semidefinite matrices. *CoRR*, abs/1107.0088, 2011.

[4] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances, 2009.

[5] Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *CoRR*, abs/0808.4134, 2008.