CSC 2421H : Graphs, Matrices, and Optimization Lecture 1 : 10 Sep 2018

Introduction

Lecturer: Sushant Sachdeva *Scribe: Dmitry Paramonov*

1 Introduction to the course

1.1 Goal

Graph algorithms have been a mainstay of algorithms since the foundation of computer science. The notion of polynomial time efficiency (the class **P**) was defined in the context of the first algorithm for matchings in general graphs. You must have seen some graph algorithms in your undergraduate algorithms class.

When you think of algorithms on graphs, you probably think of BFS / DFS – or more broadly combinatorial algorithms. However, there is another approach to designing algorithms on graphs – one that's based on matrices and continuous optimization. This approach was first taken for graph partitioning by Cheeger's inequality [AM85], which blossomed into the field of 'Spectral graph theory'. Another thread in this direction has been to incorporate numerical algorithms, and continuous optimization methods, that has become very popular over the last one and half decades since the work of Spielman and Teng [ST04].

The goal of this course will be to introduce you to the fundamentals of these fields, to make you familiar with the tools and techniques used here, and to equip you to be able to follow the latest research in this area.

1.2 Pre-requisites

Officially, the pre-requisites are undergraduate algorithms, linear algebra, probability, and some basic vector calculus. However, most importantly of all, I expect mathematical maturity. This course will be theoretical, and very mathematical in nature. We will prove several theorems, and give guarantees for our algorithms.

This is an advanced graduate course, so I expect that there will be some mathematical background that you will not be aware of, I expect you to read to make up for that background. Sometimes, I will point you to material that I expect you to read before coming to class. I expect the material to be less than 2 hrs of reading per week. If you miss reading them, it might be hard for you to follow or appreciate what I'm doing in class. At the same time, I will try to make my lectures only refer to the background material in a black-box manner.

If you want to get a sense of the level of the material, take a look at the lecture notes for my course CSC 2421H last year. The material will be substantially different but the level and the flavor of the material covered will be similar.

1.3 Logistics

- Monday 3–5pm, short break @4pm $\leq$ 5 mins

- Location: SS 2101
- Office hours: by appointment, feel free to catch me after lectures
- Email: sachdeva@cs.toronto.edu
- No class on Oct 8 (reading week)

1.4 Grading

- 20% scribe note : Scribing one lecture
- 30% grade : Take home mid-term.
- Post mid-term, you will be picking a paper(s) related to the course and reading it thoroughly. You will be expected to deliver a presentation on the paper, and prepare lecture-notes based on it.
- 50% grade : Paper presentation + making notes for the paper(s)

1.5 Outline of the Course

Here is a rough plan for the course. WARNING: This is a tentative plan, and subject to change.

1. Graphs, and basic matrices
   - Eigenvalues
   - Connections to random walks, conductance and mixing time
   - Cheeger's inequality
2. Graph Sparsification
   - Matrix concentration bounds
   - Effective-resistance / leverage scores
   - Sparsification by leverage score sampling
3. Laplacian Solvers
   - Gradient descent / Conjugate gradient
   - Preconditioning
   - A near-linear time solver
4. Optimization on graphs
   - Max-flow as linear program
   - Solving linear program approximately via continuous methods

2 Graphs and Basic Matrices

2.1 Introduction to Graphs

Any discussion of graphs must necessarily start with a definition of a graph.

Definition 2.1. *A* graph *is a pair of sets* $G = (V, E)$*.* V *is the set of* vertices*, and* E *is a set of* edges*.*

For this course, the assumption will be an undirected graph. Thus, E is a set of unordered pairs of vertices.

In common practice, there are many examples of graphs. Friendships could be modelled as a graph, with people serving as vertices and edges representing whether they are friends or not. The Internet can be modelled as a graph, wherein individual computers are vertices and edges are network connections.

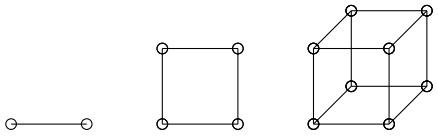
However, in this course, the focus will be on a more general case, with abstract vertices and edges.

For this purpose, vertices will usually, unless otherwise stated, be assumed to be of the form $V = [n] = \{1, \ldots, n\}$. n is usually used to denote the number of vertices in a graph, while $m = |E|$ is used to denote the number of edges.

As a simple example of a graph, consider the ring graph. The edge set is of the form $E = \big\{(i, i+1) : i \in [n-1]\big\} \cup \big\{(n, 1)\big\}$. In this case, the vertices are connected in a cycle, appearing as a ring.

This can also be modified into the path graph. In this case, the edge set is just the first, $E = \big\{(i, i+1) : i \in [n-1]\big\}$.

Another notable example is a hypercube. The vertex set for a hypercube is $V = \{0, 1\}^k$, the set of k-length strings of 0's and 1's. The set of edges is then the set of all pairs of vertices (x, y) such that x and y differ in exactly one coordinate. The hypercubes for $k = 1, 2, 3$ are sketched below.



Another useful graph is the n-star, a graph where $E = \big\{(n, i) : i \in [n-1]\big\}$. Here, all vertices are connected to one specific vertex, with no other edges in the graph.

And a complete graph is a graph where all possible edges occur, discounting self-loops, such that $E = \big\{(i, j) : i \neq j\big\}$. This is usually denoted K_n for n vertices.

2.2 Matrix Represenation of Graphs

For most purposes, algorithms to be run on graphs are run on computers. Thus, we wish for an efficient way of representing it. Matrices can be efficiently stored on a computer, so representing a graph as a matrix would be convenient.

The most straightforward way to represent a matrix on a computer is an adjacency matrix. This represents which vertices are adjacent.

Definition 2.2. *An* adjacency matrix *of a graph $G = (V, E)$ is a matrix $\mathbf{A} \in \mathbb{R}^{V \times V}$ such that*

$$\mathbf{A}(u, v) = \begin{cases} 1, & (u, v) \in E \\ 0, & \textit{otherwise} \end{cases}$$

A 1 indicates that there is an edge between these vertices, and a 0 indicates that there isn't.

This can be generalized for weighted graphs, graphs with an associated weight function $w : E \to \mathbb{R}$ by storing the weights.

$$\mathbf{A}(u, v) = \begin{cases} w(u, v), & (u, v) \in E \\ 0, & \text{otherwise} \end{cases}$$

This representation can be used to quickly check connections between weights. It allows for the testing of whether an edge is present in $O(1)$ time.

However, adjacency matrices do not have any convenient mathematical properties. Thus, while being a convenient storage format for graphs, adjacency matrices don't actually allow for any advanced matrix-based analyses.

Another similar matrix is the degree matrix. This uses the notion of the degree. The degree of a vertex v, denoted by $\deg(v)$ is the number of edges such that v is an endpoint.

Definition 2.3. *A* degree matrix *of a graph $G = (V, E)$ is a diagonal matrix $\mathbf{D} \in \mathbb{R}^{V \times V}$ such that*

$$\mathbf{D}_{u,v} = \begin{cases} 0, & u \neq v \\ \deg(u), & u = v \end{cases}$$

For this matrix, in the case of a weighted graph, the values are the weighted degree, the sum of the weights of all edges ending in this vertex.

However, this matrix also doesn't allow for convenient analysis. In fact, the original graph cannot be reliably reconsructed from it.

Thus, another matrix is introduced, with more convenient properties.

Definition 2.4. *The* Laplacian *of a graph $G = (V, E)$ is the unique symmetric matrix $\mathbf{L}_G \in \mathbb{R}^{V \times V}$ such that*

$$x^T \mathbf{L}_G x = \sum_{(u,v) \in E} \big(x(u) - x(v)\big)^2$$

For a weighted graph, the quadratic terms are rescaled by the weight.

$$x^T \mathbf{L}_G x = \sum_{(u,v) \in E} w(u,v) \big(x(u) - x(v)\big)^2$$

However, this only defines the Laplacian implicitly, and we wish to find an explicit way of computing it.

Thus, let us begin with a single edge. The graph G_{uv} has some amount of vertices, but has only one edge, between the vertices u and v. Let us attempt to compute the Laplacian for this, $\mathbf{L}_{uv}$.

$$\begin{aligned} x^T \mathbf{L}_{uv} x &= \big(x(u) - x(v)\big)^2 \\ &= x(u)^2 + x(v)^2 - 2x(u)\,x(v) \\ x^T \mathbf{L}_{uv} x &= \sum_{w,z \in V} \mathbf{L}_{uv}(w,z)\, x(w)\, x(z) \end{aligned}$$

Because we require the $x(u)^2$ and $x(v)^2$ coefficients to be 1, we see that the u,u entries and v,v entries must be 1. And therefore, the u,v and v,u entries must sum to -2. But by symmetry, we can thus see their exact values.

$$\therefore \mathbf{L}_{uv} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

We can also rewrite it another way.

$$\begin{aligned}
&\text{Let } \vec{\chi}_u \in \mathbb{R}^n \\
\vec{\chi}_u(w) &= \begin{cases} 1, & w = u \\ 0, & \text{otherwise} \end{cases} \\
x^T \mathbf{L}_{uv} x &= \left(x(u) - x(v)\right)^2 \\
&= \left(\left(\vec{\chi}_u - \vec{\chi}_v\right)^T x\right)^2
\end{aligned}$$

We can thus expand the thing in the middle.

$$\begin{aligned}
\left(\vec{\chi}_u - \vec{\chi}_v\right)(w) &= \begin{cases} 1, & w = u \\ -1, & w = v \\ 0, & \text{otherwise} \end{cases} \\
x^T \mathbf{L}_{uv} x &= \left(\left(\vec{\chi}_u - \vec{\chi}_v\right)^T x\right)^2 \\
&= \left(\left(\vec{\chi}_u - \vec{\chi}_v\right)^T x\right)^T \left(\left(\vec{\chi}_u - \vec{\chi}_v\right)^T x\right) \\
&= x^T \left(\vec{\chi}_u - \vec{\chi}_v\right)\left(\vec{\chi}_u - \vec{\chi}_v\right)^T x \\
\mathbf{L}_{uv} &= \left(\vec{\chi}_u - \vec{\chi}_v\right)\left(\vec{\chi}_u - \vec{\chi}_v\right)^T \\
&= \begin{pmatrix} 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & -1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}
\end{aligned}$$

Thus, in the single edge case, we can extract our $\mathbf{L}_{uv}$ explicitly. Note that this only concerns the submatrix involving u and v. For all other vertices, as they have no edges, all coefficients are 0.

We can also find a more general case of the Laplacian. Note that the quadratic form is necessarily linear in the edges. Thus, we can rederive the full $\mathbf{L}_G$ from that property.

$$\begin{aligned}
x^T \mathbf{L}_G x &= \sum_{(u,v)\in E} w(u,v) \left(x(u) - x(v)\right)^2 \\
&= \sum_{(u,v)\in E} w(u,v) \left(x^T \mathbf{L}_{uv} x\right) \\
&= x^T \left(\sum_{(u,v)\in E} w(u,v) \mathbf{L}_{uv}\right) x \\
\mathbf{L}_G &= \sum_{(u,v)\in E} w(u,w) \mathbf{L}_{uv}
\end{aligned}$$

We also claim that the following is another definition of the Laplacian, but the proof is left as an exercise to the reader.

$$\mathbf{L}_G = \mathbf{D} - \mathbf{A}$$

2.3 The Spectral Theorem

Now that we can represent our graphs as matrices, the obvious question becomes as to what we can do with this. Is there some information we can extract from it while it in matrix form which we could not easily extract earlier?

It turns out that we can. But to demonstrate this, we first need to cover the Spectral Theorem.

Definition 2.5. *For a given matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$, $\vec{\psi} \in \mathbb{R}^n$ is an eigenvector of M with $\lambda \in \mathbb{R}$ as an eigenvalue if $\mathbf{M}\vec{\psi} = \lambda\vec{\psi}$, under the condition that $\vec{\psi} \neq 0$.*

Note that not all matrices have eigenvalues or eigenvectors. However, square symmetric matrices have very interesting properties, which are given by the Spectral Theorem.

Theorem 2.6. *If $\mathbf{M}$ is a symmetric $n \times n$ matrix, then there exist $\vec{\psi}_1, \ldots, \vec{\psi}_n \in \mathbb{R}^n$ which are mutually orthonormal and such that each is an eigenvector of $\mathbf{M}$, with eigenvalues $\lambda_1, \ldots, \lambda_n$, respectively.*

Furthermore,

$$\mathbf{M} = \sum_i \lambda_i \vec{\psi}_i \vec{\psi}_i^T$$

Note that these eigenvalues are not necessarily distinct. You can have duplicates. But if you count them, taking into account multiplicity, we see that there are n total.

Furthermore, our orthonormal eigenvectors form a basis of the space, so we can write any vector as a linear combination of them.

As an example, consider a matrix $\mathbf{U}$, with each column being a different $\vec{\psi}_i$.

$$\mathbf{U} = \begin{pmatrix} \vec{\psi}_1 & \cdots & \vec{\psi}_n \end{pmatrix}$$

Note that we thus get that $\mathbf{U}^T\mathbf{U} = \mathbf{I}_{n \times n}$, the identity. And by properties of orthonormal matrices, we get that $\mathbf{U}\mathbf{U}^T = \mathbf{I}_{n \times n}$ And we can thus rewrite $\mathbf{M}$. Let $\mathbf{\Lambda}$ be the diagonal matrix with $\lambda_1, \ldots, \lambda_n$ on the diagonal. In that case, $\mathbf{M}$ gets a very convenient representation.

$$\mathbf{M} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$$

We also know that λ is an eigenvalue of $\mathbf{M}$ if and only if $\det(x\mathbf{I} - \mathbf{M}) = 0$ has λ as a solution.

2.4 Spectral Theorem on Graphs

Now, let us apply the above on graphs.

Let us begin with a single edge.

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

$$\mathbf{D} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$
$$\mathbf{L} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

We can now find eigenvectors for this.

$$\mathbf{L}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$\vec{\psi}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$
$$\lambda_1 = 0$$

From this, you can try to construct another one, by taking an orthogonal vector.

$$\vec{\psi}_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}$$
$$\lambda_2 = 2$$

You thus have a decomposition. You can also find the latter one just by recalling the $\vec{\chi}$ form of $\mathbf{L}_{uv}$, from which you can see that $\mathbf{L}_{uv} = (\vec{\chi}_u - \vec{\chi}_v)(\vec{\chi}_u - \vec{\chi}_v)^T$. Thus, by starting from the final step of the Spectral Theorem, we see that $(\vec{\chi}_u - \vec{\chi}_v)$ must be an eigenvector, which actually gives us exactly $\vec{\psi}_2$.

We can also try to calculate the decomposition of the complete graph.

$$\mathbf{D} = \begin{pmatrix} n-1 & 0 & \cdots & 0 \\ 0 & n-1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & n-1 \end{pmatrix}$$
$$\mathbf{A} = \begin{pmatrix} 0 & -1 & \cdots & -1 \\ -1 & 0 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & 0 \end{pmatrix}$$
$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$
$$= \begin{pmatrix} n-1 & -1 & \cdots & -1 \\ -1 & n-1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & n-1 \end{pmatrix}$$

$$= n\mathbf{I} - \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

This latter matrix is usually denoted by $\mathbf{J}$.

$$= n\mathbf{I} - \mathbf{J}$$

We can also express this using the all-ones vector, $\vec{\mathbf{1}}(v) = 1 \forall v \in V$.

$$= n\mathbf{I} - \vec{\mathbf{1}}\vec{\mathbf{1}}^T$$

Now, we can see that $\vec{\mathbf{1}}\vec{\mathbf{1}}^T$ has 1 eigenvalue of n, as before, and $n-1$ eigenvalues of 0, since this is a matrix of rank 1.

However, note that if λ is an eigenvalue of $\mathbf{M}$, then $n - \lambda$ is an eigenvalue of $n\mathbf{I} - \mathbf{M}$, and vice versa.

And thus, $n\mathbf{I} - \vec{\mathbf{1}}\vec{\mathbf{1}}^T$ has $n-1$ eigenvalues of n and 1 eigenvalue of 0.

Thus, we get the eigenvalues of K_n.

We also get some nice properties.

Lemma 2.7. *For any Laplacian, the eigenvalues are all non-negative.*

To see this, consider $\vec{\psi}_i$ and λ_i.

$$\mathbf{L}\vec{\psi}_i = \lambda_i \vec{\psi}_i$$
$$\vec{\psi}_i^T \mathbf{L}\vec{\psi}_i = \lambda_i \vec{\psi}_i^T \vec{\psi}_i$$

We know by orthonormality that the vector product must be non-negative.

$$\vec{\psi}_i^T \vec{\psi}_i \geq 0$$

But we also know by quadratic forms that the left side must also be non-negative.

$$\vec{\psi}_i^T \mathbf{L}\vec{\psi}_i \geq 0$$
$$\therefore \lambda_i \geq 0$$

2.5 Conductance

Lemma 2.8. *Every Laplacian has an eigenvalue of* 0.

To see this, consider $\mathbf{L}_G\vec{\mathbf{1}}$. We know that due to the quadratic form,

$$\vec{\mathbf{1}}^T\mathbf{L}_G\vec{\mathbf{1}} = 0$$

Alternatively, we can consider the expanded form.

$$\begin{aligned}\mathbf{L}_G\vec{\mathbf{1}} &= (\mathbf{D} - \mathbf{A})\,\vec{\mathbf{1}} \\ &= 0\end{aligned}$$

This holds due to the sum over the rows in A giving exactly the degree for that row.

But we also get some interesting properties about connectivity.

Lemma 2.9. *If G is connected, and $\mathbf{L}_G\vec{x} = 0$ with $x \neq 0$, then $\vec{x}$ must be of the form $c\vec{\mathbf{1}}$. The proof follows.*

$$\begin{aligned}\mathbf{L}_G\vec{x} &= 0 \\ \vec{x}^T\mathbf{L}_G\vec{x} &= 0 \\ \sum_{(u,v)\in E} \big(x(u) - x(v)\big)^2 &= 0 \\ \therefore \forall (u,v) \in E, \quad x(u) - x(v) &= 0 \\ \forall (u,v) \in E, \quad x(u) &= x(v)\end{aligned}$$

In general, this means that every connected component in your graph must have the same value, with the entire graph getting the same value in the connected case.

This means that for a connected graph, the only 0-eigenvalued eigenvector is $\vec{\mathbf{1}}$. Which means that all other eigenvalues are non-zero.

Lemma 2.10. *If G has k connected components, then $\mathbf{L}_G$ has k eigenvectors with eigenvalue* 0*, and all others having a greater eigenvalue.*

This is left as an exercise to the reader.

References

[AM85] Noga Alon and V. D. Milman. "λ_1, Isoperimetric inequalities for graphs, and superconcentrators". In: *J. Comb. Theory, Ser. B* 38.1 (1985), pp. 73–88 (cit. on p. 1).

[ST04] Daniel A. Spielman and Shang-Hua Teng. "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems". In: STOC. 2004, pp. 81–90. ISBN: 1-58113-852-0. DOI: 10.1145/1007352.1007372 (cit. on p. 1).