Spectral Graph Theory

Derek Chen[1] Advay Goel[2]

December 2021

[1]Student
Belmont High School

[2]Student
Acton-Boxborough Regional High School

Graph Theory Fundamentals

Graphs

Definition of Graph

A **graph** is a set of vertices V that are connected by a set of edges E with a function ψ that maps edges to unordered pairs of vertices.

Graphs

Definition of Graph

A **graph** is a set of vertices V that are connected by a set of edges E with a function ψ that maps edges to unordered pairs of vertices.

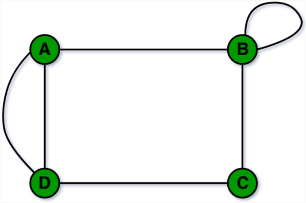


Figure 1: An undirected graph

(Image from https://brilliant.org/wiki/graph-theory/)

Graphs

Definition of Directed Graph

A **directed graph** is a graph where each edge represents an ordered pair of vertices (i.e. has **orientation**).

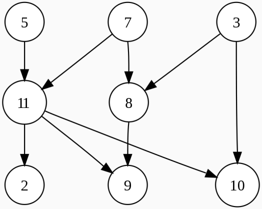


Figure 2: A directed graph with 7 vertices and 9 edges

(Image from https://en.wikipedia.org/wiki/Directed_graph)

Walks on Graphs

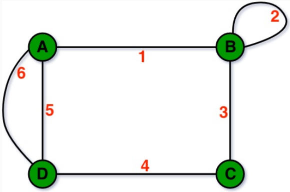
Definition of Walk

A **walk** on a graph G consists of an alternating sequence

$$V_1 E_1 V_2 E_2 \cdots E_n V_{n+1}$$

where each V_i is a vertex and E_i is an edge connecting V_i and V_{i+1}.

Example:



A 1 B 2 B 3 C – length 3

D 5 A 6 D – length 2, closed

Counting Walks On Graphs

From here, a natural question arises. For an arbitrary graph G, is there a way to count the number of walks of length ℓ?

Linear Algebra Fundamentals

Eigenvalues and Eigenvectors

Definition of Eigenvector and Eigenvalue

An **eigenvector** of a matrix A is a vector which, when multiplied by A, gives a scalar multiple of itself. The scalar multiple is called the corresponding **eigenvalue.**

This relationship can be modeled by the following equation:

$$Ax = \lambda x$$

where x is the eigenvector and λ is the eigenvalue.

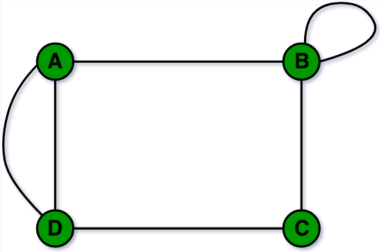
The Adjacency Matrix

Definition of Adjacency Matrix

Given a graph G with n vertices, the **adjacency matrix** of G, denoted as $A(G)$, is an $n \times n$ matrix whose (i, j)-entry a_{ij} is equal to the number of edges connecting v_i to v_j.

The Adjacency Matrix

Example:



$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 2 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 0 \end{bmatrix}$$

In this example, we denoted A as v_1, B as v_2, C as v_3, and D as v_4.

Properties of the Adjacency Matrix

The adjacency matrix:

- is symmetric
- has real eigenvalues
- has trace equal to the number of loops in G

Counting Walks on Graphs

Theorem 1.1

For a graph G, the number of walks of length ℓ for $\ell \geq 1$ that begin at vertex v_i and end at vertex v_j is the (i, j)-entry of $A(G)^\ell$.

Proof:

By the principles of matrix multiplication, the (i, j)-entry of $A(G)^\ell$ is the sum of $a_{ii_1} a_{i_1 i_2} \dots a_{a_{\ell-1} a_j}$ which counts the number of paths of length ℓ which pass through the vertices $v_i, v_{i_1}, v_{i_2}, \dots, v_{i_{\ell-1}}, v_j$ over all such combinations of $v_i, v_{i_1}, v_{i_2}, \dots, v_{i_{\ell-1}}, v_j$. $\square$

Corollary 1.2

For a graph G, let $\lambda_1, \lambda_2, \ldots, \lambda_p$ be the eigenvalues of $A(G)$. For each i, j, there exist real $c_1, c_2, \ldots, c_p$ such that for all ℓ,

$$\left(A(G)^\ell\right)_{ij} = c_1\lambda_1^\ell + c_2\lambda_2^\ell + \cdots + c_p\lambda_p^\ell.$$

Proof:

Let U be the matrix whose columns are orthonormal eigenvectors of $A(G)$, $u_1, u_2, \ldots, u_p$. Then we have $A(G)^\ell = UD^\ell U^{-1}$ where D is the diagonal matrix of the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_p$. Because U is orthogonal, $U^{-1} = U^T$ and simplifying gives us $(A(G)^\ell)_{ij} = \sum_k u_{ik}\lambda_k^\ell u_{jk}$ so $c_k = u_{ik}u_{jk}$. □

Counting Closed Walks on Graphs

Corollary 1.3

For a graph G, let $\lambda_1, \lambda_2, \ldots, \lambda_p$ be the eigenvalues of $A(G)$. Then the number of closed walks in G of length ℓ is $\lambda_1^\ell + \lambda_2^\ell + \cdots + \lambda_p^\ell$.

Proof:

Because closed walks begin and end at the same vertex, the number of closed walks of length ℓ is just the sum of the diagonal (trace) of $A(G)^\ell$. Because the trace of a matrix is the sum of its eigenvalues and the eigenvalues of $A(G)^\ell$ are $\lambda_1^\ell, \lambda_2^\ell, \ldots, \lambda_p^\ell$, corollary 1.3 follows. $\square$

Counting Closed Walks on the Complete Graph

Corollary 1.4

The number of closed walks on the complete graph K_p from vertex v_i to itself is

$$(A(K_p)^\ell)_{ii} = \frac{1}{p}\left((p-1)^\ell + (p-1)(-1)^\ell\right).$$

Proof:

The adjacency matrix of K_p has 0's on the diagonal and 1's everywhere else. The eigenvalues of $A(K_p)$ are $p-1$ and -1 with a multiplicity of $p-1$. We divide by p for an individual vertex v_i due to symmetry. □

The Matrix-Tree Theorem

Trees

Definition of Path

A **path** is a walk with no repeated vertices.

Definition of Tree

A **tree** is an undirected graph such that any two vertices are connected by *exactly* one path.

Note that trees must also have no double edges as those would be cycles of length 2. A tree on n vertices has $n - 1$ edges.

Spanning Trees

Definition of Spanning Tree

A **spanning tree** of a graph G is a tree that has its vertices equal to the vertices of G and its edges among the edges of G.

Example: Examples of spanning trees for the graph below include *abc*, *bde*, and *ace*. *ab* is not spanning and *acde* is not a tree.

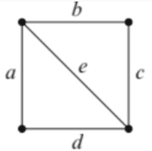


Figure 3: Complete Graphs

(Image from Algebraic Combinatorics by Richard Stanley)

Counting Spanning Trees

Definition of Complexity

The **complexity** of a graph G, denoted $\kappa(G)$, is the number of spanning trees of G.

The goal of the Matrix-Tree theorem is to determine $\kappa(G)$.

Laplacian Matrix

Definition of Laplacian Matrix

The **Laplacian matrix** $L(G)$ of a graph G with p vertices is the $p \times p$ matrix whose (i, j)-entry L_{ij} is determined by:

$$L_{ij} = \begin{cases} -m_{ij} & \text{if } i \neq j \text{ and there are } m_{ij} \text{ edges between } v_i \text{ and } v_j \\ \deg(v_i) & \text{if } i = j \end{cases}$$

where $\deg(v_i)$ is the number of edges incident to v_i.

Note that L is a symmetric matrix.

Incidence Matrix

For a graph G with p vertices and q edges, we choose an orientation $\mathfrak{o}$ where each edge e has an initial vertex u and a final vertex v.

Definition of Incidence Matrix

The **incidence matrix** $M(G)$ of a graph G with respect to orientation $\mathfrak{o}$ is the $p \times q$ matrix whose (i, j)-entry is

$$M_{ij} = \begin{cases} -1 & \text{if edge } e_j \text{ has initial vertex } v_i \\ 1 & \text{if edge } e_j \text{ has final vertex } v_i \\ 0 & \text{else} \end{cases}$$

.

Incidence and Laplacian Matrices

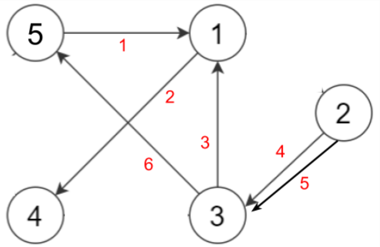


Figure 4: Red numbers represent edges

$$M(G) = \begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 & -1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad L(G) = \begin{bmatrix} 3 & 0 & -1 & -1 & -1 \\ 0 & 2 & -2 & 0 & 0 \\ -1 & -2 & 4 & 0 & -1 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & -1 & 0 & 2 \end{bmatrix}$$

The Matrix-Tree Theorem

The Matrix-Tree Theorem

Let G be a finite connected graph without loops with laplacian matrix $L(G)$. Let L_0 denote L with the last row and column removed. Then,

$$\det L_0 = \kappa(G).$$

We will now devote the rest of this presentation to proving the Matrix-Tree Theorem.

Lemma 2.1

We have $MM^T = L$.

Proof:

Pick arbitrary vertices $v_i, v_j \in V(G)$. Then,

$$\left(MM^T\right)_{ij} = \sum_{e_k \in E(G)} M_{ik} \left(M^T\right)_{kj} = \sum_{e_k \in E(G)} M_{ik} M_{jk}.$$

If $i \neq j$, then in order for this product to not equal 0, we need e_k to connect v_i and v_j. In that case, one of M_{ik} and M_{jk} equals 1 and the other is -1, so their product will always be -1. Since we sum over all edges, $\left(MM^T\right)_{ij} = -m_{ij} = L_{ij}$.
If $i = j$, then for the product to not equal 0, e_k must pass through $v_i = v_j$, in which case the product will be 1. So, $\left(MM^T\right)_{ij} = \deg(v_i) = L_{ij}$, proving the lemma.

The Binet-Cauchy Theorem

Theorem 2.2 (The Binet-Cauchy Theorem)

Let A be an $m \times n$ matrix and B be an $n \times m$ matrix. If $m > n$, then $\det(AB) = 0$. If $m \leq n$, then:

$$\det(AB) = \sum_S (\det A[S])(\det B[S])$$

where the sum goes through all m-element subsets S of $\{1, 2, \cdots n\}$.

Reduced Incidence Matrix

Definition of the Reduced Incidence Matrix

Given a graph G and its incidence matrix $M(G)$, the reduced incidence matrix $M_0(G)$ is formed by removing the last row of $M(G)$.

Note that $M_0(G)$ has $p - 1$ rows and q columns, so the number of rows equals the number of edges in a spanning tree of G.

Reduced Incidence Matrix

In the next slide, we discuss the determinants of all $(p-1) \times (p-1)$ submatrices N of M_0 which are formed as such:

1. Choose a set $X = \{e_{i_1} \cdots e_{i_{p-1}}\}$ of $p-1$ edges of G
2. Take all columns of M_0 indexed by $S = \{i_1 \cdots i_{p-1}\}$.

The Determinant of the Square Submatrix

Lemma 2.3

Let X be a set of $p-1$ edges of G. If X does not form the set of edges of a spanning tree, then the corresponding square submatrix N has determinant 0. Otherwise $\det N = \pm 1$.

The Matrix Tree Theorem

The Matrix-Tree Theorem

Let G be a finite connected graph without loops with Laplacian matrix $L(G)$. Let L_0 denote L with the last row and column removed. Then,

$$\det L_0 = \kappa(G).$$

The Matrix Tree Theorem

Proof: By Lemma 2.1, since $L = MM^T$, $L_0 = M_0 M_0^T$. Hence, by the Binet-Cauchy Theorem (Theorem 2.2), we obtain:

$$\det L_0 = \sum_S (\det M_0[S])(\det M_0^T[S]) = \sum_S (\det M_0[S])^2$$

where S ranges through all $(p-1)$-element subsets of the edges of G. By Lemma 2.3, det $M_0[S] = \det N = \pm 1$ if S forms the set of edges of a spanning tree of G and is 0 otherwise. Since we take the square, the sum adds 1 for each spanning tree and 0 otherwise. Hence, the sum equals $\kappa(G)$, proving the Matrix-Tree Theorem.

Closing Remarks

Acknowledgements

We'd like to thank

- Luis Kumanduri for his mentorship, clear explanations, and support
- Pavel Etingof, Tanya Khovanova, Slava Gerovitch, and the rest of the MIT PRIMES program for making this reading group possible
- Our parents for facilitating our participation in this program

Thank you! Any questions?