Graph Inference with Effective Resistance Queries

Huck Bennett   HUCK.BENNETT@COLORADO.EDU
University of Colorado Boulder

Mitchell Black   M4BLACK@UCSD.EDU
University of California San Diego

Amir Nayyeri   AMIR.NAYYERI@OREGONSTATE.EDU
Oregon State University

Evelyn Warton   WARTON.E@OREGONSTATE.EDU
Oregon State University

Editors: Matus Telgarsky and Jonathan Ullman

Abstract

The goal of *graph inference* is to design algorithms for learning properties of a hidden graph using queries to an oracle that returns information about the graph. Graph reconstruction, verification, and property testing are all special cases of graph inference.

In this work, we study graph inference using an oracle that returns the *effective resistance* (ER) distance between a given pair of vertices. Effective resistance is a natural notion of distance that arises from viewing graphs as electrical circuits, and has many applications. However, it has received little attention from a graph inference perspective. Indeed, although it is known that any n-vertex graph can be uniquely reconstructed by making all $\binom{n}{2} = \Theta(n^2)$ possible ER queries, very little else is known. We address this and show a number of fundamental results in this model, including:

1. $O(n)$-query algorithms for testing whether a graph is a tree; deciding whether two graphs are equal assuming one is a subgraph of the other; and testing whether a given vertex (or edge) is a cut vertex (or cut edge).

2. Property testing algorithms, including for testing whether a graph is vertex-biconnected and whether it is edge-biconnected. We also give a reduction that shows how to adapt property testing results from the well-studied bounded-degree model to our model with ER queries. This yields ER-query-based algorithms for testing k-connectivity, bipartiteness, planarity, and containment of a fixed subgraph.

3. Graph reconstruction algorithms. We highlight two k-query (provably minimal) algorithms for recovering an entire graph with k missing edge weights: an exact algorithm with an exponential running time for unweighted graphs, and an approximate numerical algorithm with polynomial running time for weighted graphs. We also give a simple, $O(k^2)$-query, polynomial-time algorithm for this problem. Additionally, we give an algorithm for reconstructing a graph from a low-width tree decomposition.

We additionally compare the relative power of ER queries and shortest path queries, which are closely related and better studied. Interestingly, we show that the two query models are incomparable in power.

Keywords: Graph theory, effective resistance, graph inference, property testing, learning algorithms

1. Introduction

Let $G = (V, E)$ be a hidden graph with n vertices, and let $d : V \times V \to \mathbb{R}^{\geq 0}$ be a metric on G. We study *graph inference* problems, where the goal is to reconstruct G or deduce properties of G using a limited number of queries to d. In this work, we focus on the case where d is the *effective resistance* (ER) metric on G.

Graph inference using the ER metric and its close cousin, the *shortest path* (SP) metric, arises naturally in many applications. Inferring a graph from SP queries has been widely studied in the context of network discovery (Beerliova et al., 2006), where it is used to map unknown regions of the internet. It is also applied in the reconstruction of evolutionary trees (Hein, 1989; King et al., 2003; Reyzin and Srivastava, 2007b), where understanding evolutionary relationships based on pairwise distances is crucial. On the other hand, ER-based inference is particularly relevant in the study of social network privacy (Aggarwal, 2014; Hoskins et al., 2018), where the goal is to assess the potential of revealing hidden network features through random-walk-based queries.

Graph inference via SP queries has also been explored extensively from a theoretical perspective, leading to notable algorithmic and lower bound results. For instance, bounded-degree graphs can be reconstructed using $\widetilde{O}(n^{3/2})$ queries (Kannan et al., 2018), and bounded-degree almost-chordal graphs can be reconstructed using $O(n \log n)$ queries (Bastide and Groenland, 2023). On the other hand, $\Omega(n^2)$ queries are needed to even reconstruct trees if they are not bounded degree (Reyzin and Srivastava, 2007a). However, the study of graph inference via ER queries is still in its early stages. Beyond the $\Theta(n^2)$-query algorithm that guarantees full reconstruction given all pairwise ER distances (see (Wittmann et al., 2009; Spielman, 2012; Hoskins et al., 2018)), little is known that comes with theoretical guarantees. Moreover, the quadratic query complexity needed to reconstruct a whole graph is impractical for large graphs and when the ability to make ER measurements is limited. The focus of this work is to address the corresponding natural question:

> What properties of a graph can be inferred from a subquadratic number of ER queries?

We interpret this question quite generally, and study graph reconstruction, property testing, and more using ER queries.

A closely related question asks what the comparative power of SP and ER queries is in the context of machine learning on graphs. In the context of graph neural networks and graph transformers, distance measures such as SP and ER are often used as inputs to the network in the form of positional encodings (Zhang et al., 2023; Velingker et al., 2024). A primary challenge in graph neural networks is to identify which positional encodings offer the highest expressive power, enabling the model to detect the most properties of a graph. For example, graph transformers using ER as a positional encoding can determine which vertices are cut vertices, while graph transformers using SP cannot (Zhang et al., 2023). While there have been several works aiming to understand and compare the relative expressive power of SP and ER encodings (Zhang et al., 2023; Black et al., 2024; Velingker et al., 2024), the capabilities of SP or ER remain only partially understood. A deeper understanding of their potential could lead to better feature selection strategies and improved learning efficiency.

1.1. Our Contribution

In this paper, we study a number of fundamental graph inference problems using ER queries. To begin with, we demonstrate that with $O(n)$ ER queries, the following tasks can be performed:

(1) Determining whether a graph is a tree (Theorem 9).

(2) Determining whether two graphs are identical, assuming one is a subgraph of the other. In other words, we can detect if some edges have been added to the graph or if some edges have been deleted, but not both (Theorem 11).

(3) Deciding whether a given vertex is a cut vertex (Theorem 12).

(4) Deciding whether a pair of vertices are adjacent with a cut edge (Theorem 14).

Furthermore, we explore (approximate) *property testing* of graphs, where our algorithms can distinguish between the case where the graph satisfies a given property and the case where it is ε-far from satisfying the property, meaning that at least εm edge modifications are required to satisfy the property. In this setting, we present the following results:

(1) Determining whether a graph is vertex-biconnected or ε-far from being vertex-biconnected with $O(n/\varepsilon)$ ER queries (Theorem 15).

(2) Determining whether a graph is edge-biconnected or ε-far from being edge-biconnected with $O(n/\varepsilon^2 + 1/\varepsilon^4)$ ER queries (Theorem 18).

(3) Showing that for any property that can be tested in $f(\varepsilon, n)$ time in the well-studied bounded-degree model, there is an algorithm to test it for bounded-treewidth graph with $f(\varepsilon, n) \cdot n$ ER queries (Theorem 19). As a result, we obtain property testers that require a subquadratic number of ER queries for bounded-treewidth graphs, addressing a range of problems such as the inclusion of a fixed subgraph, k-connectivity, bipartiteness, the presence of long cycles, and any minor-closed property (e.g., planarity) (Goldreich, 2017).

We remark that the running time of our property testing algorithms for the latter two results can be reduced by a factor of n, making them dependent only on ε, given a stronger ER query model. In this model, queries return vertices in the order of their distance from a vertex v, allowing the algorithm to request only the closest k vertices to v. This stronger query model is natural, as in most applications, closer vertices are often accessible with less effort, for example if close vertices are sampled by a random walk.

In addition to our property inference algorithms, we analyze the relative power of ER and SP query models. Interestingly, we show that the two query models are incomparable, i.e., neither model is strictly stronger than the other for all tasks. Specifically, we demonstrate the following results:

(1) Determining whether a graph is a clique can be achieved with $O(n)$ ER queries, whereas it requires $\Omega(n^2)$ SP queries (Theorem 27).

(2) Checking whether two given vertices are adjacent can be achieved with a single SP query, whereas it requires $\Omega(n)$ ER queries (Theorem 28).

Finally, we study the problem of graph reconstruction using ER queries. Specifically, we give an algorithm for each of the following tasks:

(1) Given a width-k tree decomposition of a graph G, reconstructs G using $O(k^2 n)$ ER queries (Theorem 31).

(2) Given an adjacency matrix A of a graph G with k missing entries, recovers A using $O(k^2)$ queries and runs polynomial time (Theorem 32).

(3) Given an adjacency matrix A of a graph G with k missing entries, recovers A using k queries and runs in exponential time (Theorem 37).

Our reconstruction algorithm in Theorem 37 uses techniques from convex analysis and relies on a highly nontrivial structural property of ER distances, which allows us to reduce the number of queries. It remains an open question whether a polynomial-time algorithm can be achieved with a subquadratic number of queries, i.e., whether it is possible to combine the strengths of Theorems 32 and 37.

1.2. Related work

Graph inference with different query models. Graph inference has been studied under various models, with edge detection and edge counting being two prominent approaches motivated by applications in biology. In these models, queries allow one to check whether an induced subgraph contains any edges or to determine the number of edges in the subgraph (Alon et al., 2004; Angluin and Chen, 2008; Alon and Asodi, 2005; Bouvel et al., 2005). Reyzin and Srivastava (2007a) provides an extensive survey of results within these models, as well as the shortest path (SP) query model. Notably, they show that reconstructing a hidden tree requires at least $\Omega(n^2)$ SP queries, establishing that the bounded-degree assumption is necessary for obtaining nontrivial results using SP queries.

Kannan et al. (2018) demonstrated that bounded-degree graphs can be fully reconstructed using $\widetilde{O}(n^{3/2})$ SP queries. They further showed that bounded-treewidth chordal graphs and outerplanar graphs can be reconstructed with $\widetilde{O}(n \log n)$ SP queries. This result has been recently generalized to bounded-treewidth graphs without long cycles by Bastide and Groenland (2024), with additional related work by Rong, Li, Yang, and Wang (2021).

In contrast to the SP model, significantly fewer theoretical results are known for the effective resistance (ER) model. It has been established that a hidden graph can be fully reconstructed if the ER distances between all pairs of its vertices are known (Wittmann et al., 2009; Spielman, 2012; Hoskins et al., 2018). On the other hand, since ER and SP distances are equivalent for trees, it follows that reconstructing a general tree requires $\Omega(n^2)$ ER queries, and hence the known $O(n^2)$ query algorithm is tight for general graphs, as is the case for the SP model. However, in contrast to the SP model, subquadratic algorithms are not known for any family of graphs beyond bounded-degree trees.

A continuous variant of the graph reconstruction problem, known as *Calderón's inverse problem*, is to recover the conductivity of an object from measurements of current and potential on its surface. Calderón's inverse problem has been studied extensively by mathematicians (Uhlmann, 2012) and found important applications in Electrical Impedance Tomography (EIT) in medical imaging (Uhlmann, 2009) and Electrical Resistivity Tomography (ERT) in geophysics (Wikipedia contributors, 2024).

Property testing. We study the gap version of checking properties of graphs using ER queries. In this setting, the goal is to distinguish between graphs that possess a certain property and those that are "far" from having that property, using only a few ER queries. This part of our work is related to property testing of graphs in the bounded-degree model. Many properties have been studied in this model, including k-connectivity, bipartiteness, subgraph exclusion, minor exclusion, and planarity. We refer the reader to Goldreich (2017) for an extensive exposition of known results in this area. For contrast, we study property testing algorithms that rely only on resistance distance queries rather than adjacency queries.

More broadly, our work is also related to sublinear-time algorithms for finite metric spaces. Such algorithms become particularly relevant when one is interested in estimating parameters such as the average ER distance or identifying the central point in the ER metric space. Notably, these algorithms operate without reconstructing the graph or explicitly learning its topology. There is a substantial body of work on testing properties of metric spaces; see, for example, (Indyk, 1999; Parnas and Ron, 2003; Onak, 2008).

Graph machine learning. One important (albeit less direct) motivation for our work is graph machine learning. Currently, there are no polynomial-time machine learning algorithms (e.g., graph neural networks) that can completely capture the structure of a graph; that is to say, all existing methods will give the same output on some set of non-isomorphic graphs (see (Xu et al., 2019) for an example). This means that no graph learning algorithm captures all properties of a graph.

As an imperfect solution, one approach is compute some quantities associated with the graph and use these as input to the machine learning algorithm. For example, one could compute the effective resistance between some pairs of vertices (as in (Zhang et al., 2023; Velingker et al., 2024)) and use this as an input to a neural network. These graph quantities are called *positional encodings*. Since the use of positional encodings is known not to completely capture the topology of a graph, researchers instead ask which properties of a graph these encodings do capture. For example, effective resistance increases the expressive power of message-passing neural networks (Velingker et al., 2024) and transformers (Vaswani, 2017) using effective resistance as a positional encoding can determine which vertices are cut vertices (a property neither message-passing neural networks nor shortest-path distance can detect) (Zhang et al., 2023) and which edges are cut edges (Black et al., 2024). At a high level, the goal is to understand the power of effective resistance in deducing graph properties when used as a positional encoding in a graph neural network. This question becomes particularly relevant to our work when it focuses on identifying a small subset of effective resistances that are sufficiently informative for the task at hand, thereby reducing computational costs.

2. Preliminaries

2.1. Graph Laplacian and Effective Resistance

Let $G = (V, E, w)$ be an undirected graph with edge weights $w : E \to \mathbb{R}^{\geq 0}$. We denote the number of vertices as $n := |V|$ and the number of edges as $m := |E|$. The *graph Laplacian* is the $n \times n$ matrix defined $L := D - A$, where the *weighted degree matrix* is the diagonal matrix with entries $D_{u,u} = \sum_{(u,v) \in E} w(u,v)$ and the *weighted adjacency matrix* is the matrix with entries $A_{u,v} = w(u,v)$ if $(u,v) \in E$ and $A_{u,v} = 0$ otherwise. Alternatively, the graph Laplacian is $L = \partial W \partial^T$, where W is the $m \times m$ diagonal weight matrix with entries $W_{e,e} = w(e)$ and ∂ is the

$n \times m$ *signed incidence matrix* (corresponding to the *boundary operator*), which has entries[1]

$$\partial_{v,e} := \begin{cases} 1 & \text{if } e = (v, w) \text{ for some } w , \\ -1 & \text{if } e = (u, v) \text{ for some } u , \\ 0 & \text{otherwise} . \end{cases}$$

Finally, a third way of writing the graph Laplacian is as the sum $L = \sum_{(u,v) \in E} w(u, v) L_{uv}$, where $L_{uv} = (\mathbf{1}_u - \mathbf{1}_v)(\mathbf{1}_u - \mathbf{1}_v)^T$ is the *edge Laplacian*.

Let $M \in \mathbb{R}^{m \times n}$ be a matrix. The *psuedoinverse* of M, denoted M^+, is any matrix such that the following conditions hold: (1) $AA^+A = A$, (2) $A^+AA^+ = A^+$, (3) $(AA^+)^T = AA^+$, and (4) $(A^+A)^T = A^+A$.

The *effective resistance* between a pair of nodes u and v is defined as

$$R(u, v) := (\mathbf{1}_u - \mathbf{1}_v)^T L^+ (\mathbf{1}_u - \mathbf{1}_v) , \tag{1}$$

where L^+ is the pseudoinverse of L and $\mathbf{1}_u, \mathbf{1}_v \in \mathbb{R}^n$ are the indicator vectors of u and v. One can show that R is a metric on the vertices in the graph. Interpreting graphs as electrical circuits, the effective resistance measures the resistance between u and v in the electrical network where each edge e has conductance equal to its weight, $w(e)$, giving each edge a resistance of $1/w(e)$. The *uv-potentials* are $\boldsymbol{p}_{u,v} := L^+(\mathbf{1}_u - \mathbf{1}_v)$, as these are the voltage potentials on the vertices resulting in a unit flow of current from u to v in the graph.

Notably, the kernel of any graph Laplacian L contains the span of the all-ones vector (i.e., $\text{span}(\mathbf{1}) \subseteq \ker(L)$), since the weighted degree of a vertex is equal to the sum of the weights of its incident edges. Moreover, if the graph is connected then $\ker(L) = \text{span}(\mathbf{1})$.

To see the matrix form of the relationship between L^+ and effective resistance, let J be the all-ones matrix. The matrix $I - \frac{1}{n}J$ projects any vector onto the orthogonal complement of $\text{span}(\mathbf{1})$. Its conjugation with $R \in \mathbb{R}^{n \times n}$, the matrix of all effective resistances defined by $R_{i,j} = R(i, j)$, gives L^+ up to a constant factor:

$$-\frac{1}{2}\left(I - \frac{1}{n}J\right) R \left(I - \frac{1}{n}J\right) = L^+ \tag{2}$$

This formula implies the following lemma (Wittmann et al., 2009; Spielman, 2012; Hoskins et al., 2018).

Lemma 1 *Any weighted graph with n vertices can be reconstructed with $\binom{n}{2}$ effective resistance queries.*

Due to the $O(n^\omega)$ time algorithm for obtaining a matrix's rank-normal form given by (Keller-Gehrig, 1985), the psuedoinverse of L, and hence the matrix of all effective resistances, can be obtained in $O(n^\omega)$ runtime where $\omega < 2.371339$ (Alman et al., 2025) is the matrix multiplication exponent.

Alternatively, we can characterize the effective resistance in terms of the amount of the amount of current on the edges. A *unit uv-flow* is a flow $f : E \to \mathbb{R}$ such that $\partial f = \mathbf{1}_u - \mathbf{1}_v$. The *energy* of a flow $f : E \to \mathbb{R}$ is defined to be $\mathcal{E}(f) := \sum_{e \in E} f(e)^2/w(e)$.

---

1. The choice of which of the first two cases is equal to 1 and which equal to -1 is arbitrary. The graph Laplacian will be the same for either choice, and other quantities involving the signed incidence matrix will only differ by a sign flip.

Lemma 2 (Thompson's Principle ([Thomson and Tait, 1867])) *Let $\mathcal{F}_{uv}$ be the set of unit uv-flows. The effective resistance between two vertices $u, v \in V$ is equal to the minimum energy of any unit uv-flow, $R(u, v) = \min_{f \in \mathcal{F}_{uv}} \mathcal{E}(f)$.*

The minimum energy uv-flow $f_{u,v} := \arg\min_{f \in \mathcal{F}_{uv}} \mathcal{E}(f)$ is called the *electrical uv-flow*. Thompson's principle immediately implies *Rayleigh's monotonicity law*: if the resistance values increase the effective resistance between any pair of vertices can only increase.

Lemma 3 (Rayleigh's Monotonicity Law ([Rayleigh, 1877])) *Let $G = (V, E_G, w_G)$ and $H = (V, E_H, w_H)$ be weighted graphs on the same set of vertices V. If $E_G \subseteq E_H$ and $w_G(e) \leq w_H(e)$ for all $e \in E_G$, then $R_G(u, v) \geq R_H(u, v)$ for all $u, v \in V$.*

We find the following basic properties of effective resistance useful in this paper.

Lemma 4 *Let $G = (V, E)$. Then for all $u, v \in V$,*

(1) If $(u, v) \in E$, then $R(u, v) \leq 1$. Moreover, $R(u, v) = 1$ if and only if (u, v) is a cut edge.

(2) If there is exactly one path γ between two vertices $u, v \in V$, then the $R(u, v)$ is the length of γ.

(3) The ER distance between any pair of vertices in two different edge biconnected components is at at least one.

The regularized graph Laplacian As mentioned above, the graph Laplacian is never full-rank regardless of the graph; the all-ones vector $\mathbf{1}$ is always in the kernel. The *regularized graph Laplacian* is a slightly modified version of the graph Laplacian defined as

$$\widetilde{L} := L + \frac{1}{n} \cdot J. \tag{3}$$

For a connected graph, $\widetilde{L}$ is full-rank.

Lemma 5 ([Ghosh et al.] ([2008], Equation 9)) *Let $G = (V, E)$ be a connected graph with Laplacian L. The following three facts are true of the regularized graph Laplacian:*

1. *$\tilde{L}^{-1} = L^{+} + \frac{1}{n} J$.*

2. *For all $u, v \in V$, $R(u, v) = (\mathbf{1}_u - \mathbf{1}_v)^T \tilde{L}^{-1} (\mathbf{1}_u - \mathbf{1}_v)$.*

3. *$\tilde{L}$ is positive semidefinite for any (possibly disconnected) graph. $\tilde{L}$ is positive definite if and only if G is connected*

Schur complement. The *Schur complement* of the Laplacian L of a matrix on a vertex subset U, denoted L_U, is defined as follows.[2]

---

2. The Schur complement is usually defined as $L_U := L(U, U) - L(U, \overline{U}) L(\overline{U}, \overline{U})^{-1} L(\overline{U}, U)$ (e.g., in ([Spielman], [2025], Lemma 11.5.3)), using the inverse $L(\overline{U}, \overline{U})^{-1}$ instead of the pseudoinverse. We use the pseudoinverse as it is more general while preserving the relevant properties of the Schur complement. To see why it preserves these properties, note that the submatrix $L(\overline{U}, \overline{U})$ is invertible iff there is an edge between U and $\overline{U}$. However, if there is no edge between U and $\overline{U}$, then $L(U, \overline{U}) = 0$ and $L_U = L(U, U)$. Likewise, if there are no edge between U and $\overline{U}$, then L_U is a graph Laplacian of subgraph of G induced by U, so all the properties of the Schur complement mentioned in this section hold.

$$L_U := L(U, U) - L(U, \overline{U}) L(\overline{U}, \overline{U})^+ L(\overline{U}, U) , \tag{4}$$

where $L(X, Y)$ is the submatrix of L indexed by rows in X and columns in Y for $X, Y \subseteq V$. Note that for a connected graph, $L(\overline{U}, \overline{U})$ is always full rank. It is known that the Schur complement of a Laplacian of a graph G is the Laplacian of a (possibly weighted) graph with vertex set U, which we refer to as G_U. The following lemma summarizes some properties of the Schur complement that we find helpful in this paper (Spielman, 2025).

Lemma 6 *Let $G = (V, E)$ be a weighted graph, let $U \subseteq V$, and let G_U be the Schur complement of G on U. Then the following properties hold.*

(1) For any $u, v \in U$, their effective resistance in G is equal to their effective resistance in G_U.

(2) Let $u \in U$ be such that $w_G(u, v) = 0$ for all $v \in \overline{U}$. Then, for any $u' \in U$, $w_G(u, u') = w_{G_U}(u, u')$, i.e., the neighborhood of u is identical in G and G_U.

The Schur complement matrix can be obtained by using Gaussian elimination on the graph Laplacian. Likewise, for each step of Gaussian elimination, there is a corresponding operation on the graph, with the operations corresponding to entire process of Gaussian elimination resulting in the graph G_U. For a vertex $v \in \overline{U}$, performing Gaussian elimination on the vth row and column corresponds to removing the vertex v and, for each pair of neighbors $(a, b) \in N(v) \times N(v)$, replacing its weight with $w(a, b)' := \frac{w(a,b)}{d(v)}$, the ratio between the edge weight and the degree of v.

Tree decomposition. Let $G = (V, E)$ be a graph. A *tree decomposition* is a pair $(\mathcal{B}, T)$, where $\mathcal{B}$ is a set of subsets of V called *bags* and T is a tree with vertex set $\mathcal{B}$ such that

(1) $\bigcup_{B \in \mathcal{B}} B = V$.

(2) For every $(u, v) \in E$, there exists $B \in \mathcal{B}$ such that $u, v \in B$.

(3) Let $B, B'' \in \mathcal{B}$, and suppose $B' \in \mathcal{B}$ is a bag on the unique B to B'' path in T. Then, $B \cap B'' \subseteq B'$.

We show that given a tree decomposition of a graph with bounded treewidth, it is possible to fully recover the graph using $o(n^2)$ effective resistance queries.

Graph cuts and connectedness. Let $G = (V, E)$ be a graph. A vertex $v \in V$ is a *cut vertex* if the subgraph induced by $V \setminus \{v\}$ has more connected components than G. An edge $(u, v) \in E$ is a *cut edge* if the subgraph given by $(V, E \setminus \{(u, v)\})$ has more connected components than G. A graph is *vertex biconnected* if it contains no cut vertices. A graph is *edge biconnected* if it contains no cut edges.

3. Checking graph properties

There are several properties of graphs that can be characterized by the resistance distance on a subset of the graph's vertices. One such property is whether a given vertex is a cut vertex. We give a characterization for such vertices, and use this result to solve several decision problems on graphs, as well as to give property testing algorithms for certain connectedness properties, which, in a sense, gives an approximate solution to the corresponding decision problems. This is a folklore theorem, but we include a proof as we could not find one in the literature.

Lemma 7 (Folklore) *Let $G = (V, E)$ be an undirected, unweighted graph, and let $a, b, c \in V$ be distinct vertices. Then, b is a cut vertex between a and c (that is, a and c are in different connected components of the subgraph of G induced by $V \setminus \{b\}$, $G_{|V \setminus \{b\}}$) if and only if $R(a, b) + R(b, c) = R(a, c)$.*

Proof First we will prove that the triangle inequality holds exactly equal when the middle vertex is a cut vertex. Suppose b is a cut vertex between a and c, and $f \in \mathbb{R}^{|E|}$ be the unit electrical flow. By the flow decomposition $\exists f_1, \ldots, f_k \in \mathbb{R}^{|E|}$ such that $f = f_1 + \cdots + f_k$ for some $k \leq |E|$. Moreover, for $i \leq k$, there is a simple a to c path π_i that is the support of f_i. For each path π_i, since b is a cut between a and c, it follows that b must lie on this path.

Let $V \setminus b = A \cup C$ be a disjoint partition of vertices other than b, by assumption, and denote G^A to be the subgraph of G induced by the vertex set $V \setminus C$ and G^C to be the subgraph of G induced by the set $V \setminus A$. If we restrict f_i to G^A then b will be the endpoint of each path $\pi_i|_{G^A}$. This is because c lies on the other side of the cut vertex b, and a simple path cannot traverse b twice. So it follows that this is a unit a to b flow. If a lower energy unit a to b flow existed in G^A, it could be used in place of f under this restriction, reducing the overall energy of f (this would contradict that f is the electrical flow). A lower energy unit a to b cannot exist in G but not in G^A, since everything on G^C would clearly form a circulation back to b. So restricting f to G^A gives a miminum energy a to b flow, and a similar argument gives the fact that restricting f to G^C gives a mimimum energy a to c flow.

Since the edges of G^A and G^C are disjoint it follows that $\|f|_{G^A}\|_2^2 + \|f|_{G^C}\|_2^2 = \|f\|_2^2$, or equivalently, that

$$R(a, b) + R(b, c) = R(a, c).$$

Suppose now, to show the other direction of implication, that $R(a, b) + R(b, c) = R(a, c)$. By definition this is equivalent to having

$$(\mathbf{1}_a - \mathbf{1}_b)^T L^+ (\mathbf{1}_a - \mathbf{1}_b) + (\mathbf{1}_b - \mathbf{1}_c)^T L^+ (\mathbf{1}_b - \mathbf{1}_c) = (\mathbf{1}_a - \mathbf{1}_c)^T L^+ (\mathbf{1}_a - \mathbf{1}_c).$$

With some rearranging and cancellations, we obtain the equality

$$\mathbf{1}_a L^+ (\mathbf{1}_b - \mathbf{1}_c) = \mathbf{1}_b L^+ (\mathbf{1}_b - \mathbf{1}_c).$$

Since $L^+ (\mathbf{1}_b - \mathbf{1}_c)$ gives the potentials in the electrical b to c flow, this means that a and b have equal potential in this flow. Let $A \subseteq V$ be the set of all vertices with potential equal to a, and let $C = V \setminus A$ be its complement. By the equation we obtained, we know $b \in A$. Clearly c has potential different than b, as they are distinct vertices, and the effective resistance between them, as a metric, must be non-zero. Suppose, by contradiction, that b is not a cut vertex between a and c. Then there must be an edge between A and C whose endpoint in A is a vertex other than b, call this vertex x. This would imply, by the potential difference, that some amount of flow belongs on this edge in the electrical b to c flow. But flow exiting x implies flow entering x, which implies a vertex exists with potential larger than b, the source, yielding a contradiction. ∎

3.1. Acyclicity

can be used directly to check whether a vertex is a cut vertex, but it also allows for a concise proof of the following lemma, which states that, in an unweighted graph, the set of distances to any particular vertex must be all integers, if the graph contains no cycles.

Lemma 8 *Let $G = (V, E)$ be a connected unweighted graph and $v \in V$ be any vertex. The graph G has a cycle if and only if there exists a vertex $u \in V$ such that $R(u, v)$ is not an integer.*

Proof If G is acyclic then there is a unique path between every pair of its vertices. Hence, by Item 2 of Theorem 4, the distance between every pair of vertices is an integer.

To prove the other direction, suppose G contains a cycle. Let $v \in V$ be arbitrary, and let $x \in V$ be a closest vertex to v that is contained in a cycle C. Furhter, let $(x, y) \in E$ be an edge of C (note x might be equal to v). By Item 2 of Theorem 4, $R(v, x)$ is an integer. By Item 1 of Theorem 4, $R(x, y) < 1$. So, by triangle inequality, $R(v, y) < R(v, x) + 1$. On the other hand, by Theorem 7, $R(v, y) = R(v, x) + R(x, y) > R(v, x)$. Overall, $R(v, x) < R(v, y) < R(v, x) + 1$, and $R(v, x) \in \mathbb{Z}^+$ implies that $R(v, y)$ is not integer. ∎

It is noteworthy that the characterization holds for any vertex of the graph, giving rise to our algorithm. By choosing an arbitrary vertex of our graph, and querying its pairwise distance to the remaining vertices, we are able to detect a cycle within our graph using only $n - 1$ queries.

Theorem 9 *Let $G = (V, E)$ be an undirected unweighted graph, and let $n = |V|$. There exists an algorithm that decides whether G is a tree using $n - 1$ ER queries.*

Proof Let $v \in V$ be an arbitrary vertex of G. Verifying its pairwise distances are all integers can be done using $n - 1$ ER queries by querying its distance to each other vertex $u \in V \setminus \{v\}$. If all of these ER distance are finite we conclude that G is connected. If, in addition, all of them are integers we conclude that G is acyclic by Theorem 8. If G is both connected and acyclic then it is a tree. ∎

The effective resistance, as a metric, is highly sensitive to global changes within the graph, which is why querying all of the distances from only one vertex allows for detecting cycles. This is something that is not possible using shortest path queries. In fact, the technique of querying the resistance distance of one vertex to each other vertex can be leveraged even further to detect changes in the graph, as we do in the next section.

3.2. Equality of graphs

The sensitivity of resistance distance can be felt by every vertex of the graph. Even a small local change, such as the addition of a single edge between two vertices (or even a change to a single edge's weight), has ripple effects throughout the entire graph. In fact, Rayleigh's Monotonicity Principle states that decreasing the resistance of any edge cannot cause the effective resistance between any two nodes to increase. What we show is that, for any connected graph, for any vertex s in this graph, adding an edge (or decreasing its resistance for weighted graphs) causes a strict decrease in the resistance distance between s and some other vertex. In a sense, this gives a 'strict monotonicity' property for connected graphs, wherein lowering resistance on any edge guarantees there is a pair involving every vertex for which the effective resistance on that pair decreases.

Lemma 10 *Let $G = (V, E_G, w_G)$ and $H = (V, E_H, w_H)$ be two connected n-vertex graphs such that for every pair $(a, b) \in V \times V$, $w_G(a, b) \leq w_H(a, b)$ and there exists $(u, v) \in V \times V$ such that $w_G(u, v) < w_H(u, v)$ strictly. Then for every $s \in V$ there exists a vertex $t \in V$ such that $R_H(s, t) < R_G(s, t)$.*

Proof By Rayleigh's monotonicity law, it suffices to prove the given statement for the following case: suppose that all pairs other than (u, v) have equal weights, and only $w_G(u, v) < w_H(u, v)$.

Let $s \in V$ be any vertex, and suppose without loss of generality that there exists an s to u path π that avoids v. This assumption is alright as at most one of the following conditions holds: (1) u is a cut vertex that separates s from v, and (2) v is a cut vertex that separates s from u.

Now, let $f : E_G \to \mathbb{R}$ be the electrical s to v flow in G, and let $f' : E_H \to \mathbb{R}$ be the same flow in H, i.e..

$$f'(e) = \begin{cases} f(e) & e \in E_G \\ 0 & \text{otherwise} \end{cases}$$

Note, by Thompson's law, $R_G(s, v) = \mathcal{E}(f)$, and $R_H(s, v) \le \mathcal{E}(f')$ as f' is not necessarily an electrical flow. We show by considering two cases that in fact $R_H(s, v) < \mathcal{E}(f')$.

First, we consider the case that $f(u, v) = 0$ (in particular that can happen if $w_G(u, v) = 0$.) We say that a vertex $x \in V$ is used by f' if some edge incident to x has a non-zero f' value. Let $r \in V$ be the closest vertex to u on π that is used by f' (note r can be equal to s or u). Then, let $\pi' = \pi[r, u] \circ (u, v)$ be the r to v path in H, and note that none of the edges of π' is used in f'. Also, since r is used by f' there is at least one r to v path γ' in f' that has positive f' value on all its edges. Now, we build f'' in H by sending an ε amount of flow on π', and pushing back an ε amount of flow on γ', i.e.

$$f''(e) = \begin{cases} \varepsilon & e \in \pi' \\ f'(e) - \varepsilon & e \in \gamma' \\ f'(e) & \text{otherwise.} \end{cases}$$

It follows that,

$$g(\varepsilon) = \mathcal{E}(f'') - \mathcal{E}(f') = \sum_{e \in \pi'} (w_H(e)^{-1} \cdot \varepsilon)^2 + \sum_{e \in \gamma'} \left((w_H(e)^{-1} \cdot (f'(e) - \varepsilon))^2 - (w_H(e)^{-1} \cdot f'(e))^2 \right).$$

Thus,

$$\frac{dg}{d\varepsilon} = 2\varepsilon \sum_{e \in \pi'} (w_H(e)^{-1}) - 2 \sum_{e \in \gamma'} w_H(e)^{-1} \cdot (f'(e) - \varepsilon)$$

which, when evaluated at $\varepsilon = 0$, is $-2 \sum_{e \in \gamma'} w_H(e)^{-1} \cdot f'(e) < 0$. Since $g(0) = 0$, and the derivative of g is negative at 0, for sufficiently small values of ε, $g(\varepsilon) < 0$, which means $\mathcal{E}(f'') < \mathcal{E}(f')$. But, we already know $R_H(s, v) \le \mathcal{E}(f'')$ and $R_G(s, v) = \mathcal{E}(f')$, implying, in the case that $w_G(u, v) = 0$, the proof is complete.

Second, we consider the case that $f(u, v) > 0$. In this case, we have $w_G(u, v) > 0$, as otherwise f cannot use the non-existent edge. Then trivially

$$\mathcal{E}(f) = \sum_{e \in E_G} (w_G(e)^{-1} \cdot f(e))^2 > \sum_{e \in E_G} (w_H(e)^{-1} \cdot f(e))^2 = \mathcal{E}(f')$$

concluding the proof. ■

This property of subgraphs allows for a simple algorithm with a linear query complexity that detects if one graph is equal to another, assuming its weights have monotonically increased or decreased.

Theorem 11 *Let $G = (V, E_G, w_G), H = (V, E_H, w_H)$ be n-vertex graphs, where G is visible and H is hidden. Suppose $w_G(e) \leq w_H(e)$ for every edge $e \in E_G \cup E_H$ and there exists at least one edge e' such that $w_G(e') < w_H(e')$ strictly (or, similarly, $w_G(e) \geq w_H(e)$ for every e and there exists e' such that $w_G(e') > w_H(e')$). Then, there exists an algorithm that decides whether $G = H$ using $n - 1$ ER queries.*

Proof Let $s \in V$ be an arbitrary vertex. For all $v \in V \backslash \{s\}$, our algorithm queries $R_H(s, v)$ and computes $R_G(s, v)$. We accept if $R_H(s, v) = R_G(s, v)$ for all v and reject otherwise. If $G = H$ all these ER distances are equal and the algorithm correctly accepts. On the other hand, by Theorem 10, if $G \neq H$, there will be at least one $v \in V$ for which $R_G(s, v) \neq R_H(s, v)$, thus our algorithm correctly rejects. ∎

For unweighted graphs, this implies a linear query complexity algorithm for detecting wether two graphs are equal, assuming one is a subgraph of the other.

3.3. Cut vertices

Now, we use Theorem 7 to test whether a given vertex is a cut vertex, or two given vertices belong to the same biconnected components, both with linear number of ER queries. Now, we use Theorem 7 to test whether a given vertex is a cut vertex, or two given vertices belong to the same biconnected components, both with linear number of ER queries.

Theorem 12 *Let $G = (V, E)$ be an undirected, weighted or unweighted, graph, and let $n = |V|$.*

(1) There exists an algorithm that decides whether a vertex $v \in V$ is a cut vertex using $2n - 3$ ER queries.

(2) There exists an algorithm that decides whether two vertices $a, b \in V$ are in the same biconnected component using $2n - 3$ ER queries.

Proof First, we prove (1). To decide if v is a cut vertex, our algorithm fixes some $u \in V \setminus \{v\}$, and queries $R(u, v)$. For every $w \notin \{u, v\}$, we will then query $R(u, w)$ and $R(v, w)$. Therefore, we have a total number of $2(n - 2) + 1 = 2n - 3$ queries. Our algorithm concludes that v is a cut vertex if and only if there exists a $w \in V$, $w \neq v$, such that $R(u, w) = R(u, v) + R(v, w)$. If v is a cut vertex, then removing it separates u from at least one other vertex w in G, i.e. w can be any vertex that is in a different connected component from u in $G \backslash \{v\}$. In this case, by Theorem 7, $R(u, w) = R(u, v) + R(v, w)$, and our algorithm correctly decides that v is a cut vertex. On the other hand, if v is not a cut vertex, then $R(u, w) = R(u, v) + R(v, w)$ does not hold for any w, by Theorem 7. Therefore our algorithm correctly decides that v is not a cut vertex.

Next, we prove (2). To decide if a and b belong to the same biconnected compoenent, we query $R(a, r)$ and $R(b, r)$ for every $r \in V$, hence $2n - 3$ number of ER queries. Our algorithm concludes that a and b are in different biconnected component if and only if $R(a, r) + R(r, b) = R(a, b)$ for some $r \in V$, $r \neq a, b$. If a and b are in different biconnected components, by Theorem 7, there exists an r that makes the triangle inequality tight. Therefore, our algorithm correctly decides that a and b are in different biconnected components. On the other hand, if a and b belong to the same biconnected component, the triangle inequality is not tight for any $r \in V$, by Theorem 7. Hence, our algorithm correctly decides that a and b are in the same biconnected components. ∎

3.4. Cut edges

Not only can we characterize cut vertices by resistance distance, but we can also do the same for cut edge. For a cut edge, we show that for any vertex in the graph, its resistance distance to each of the endpoints of the edge cut differ by exactly one.

Lemma 13 *Let $G = (V, E)$ be a connected undirected unweighted graph. A pair of vertices $a, b \in V$ is a cut edge if and only if $|R(a, x) - R(b, x)| = 1$ for all $x \in V$.*

Proof Suppose that $|R(a, x) - R(b, x)| = 1$ for all $x \in V$. Let A and B be sets such that $A = \{x \in V \mid R(b, x) = R(a, x) + 1\}$, i.e. vertices that are closer to a than b, and $B = \{x \in V \mid R(a, x) = R(b, x) + 1\}$, i.e. vertices that are closer to b than a. By the assumption of lemma, A and B partition V. In particular, note $a \in A$ and $b \in B$. For any $a' \in A$, we have $R(a', b) = R(a', a) + 1 = R(a', a) + R(a, b)$. So, by Theorem 7, a is a cut vertex that separates any $a' \in A\backslash\{a\}$ from b. Using the same argument, we conclude that b is a cut vertex that separates any $b' \in B\backslash\{b\}$ from a.

Now, let π be the shortest a to b path. Since for every vertex x in $V\backslash\{a, b\}$ either a separates x from b or b separates x from a, π cannot have any vertex other than a and b. So, $(a, b) \in E$. Also, $R(a, b) = |R(a, b) - R(b, b)| = 1$. So, by Theorem 4, (a, b) is a cut edge

To see the other direction, we suppose (a, b) is a cut edge. Let $A \subseteq V$ and $B \subseteq V$ be the subset of vertices in the connected component of a and b in $G\backslash\{(a, b)\}$, respectively. We know that $R(a, b) = 1$ by Theorem 4. We also know that a is a cut vertex separating A from b, and b is a cut vertex separating a from B. Therefore, by Theorem 7, for any vertex $x \in A$, $R(x, b) = R(x, a) + R(a, b) = R(x, a) + 1$. Similarly, for any vertex $x \in B$, $R(x, a) = R(x, b) + R(a, b) = R(x, b) + 1$. Since $\{A, B\}$ is a partition of V, $|R(a, x) - R(b, x)| = 1$ for all $x \in V$ as desired. ∎

To decide if a pair of vertices $a, b \in V$ are adjacent via a cut edge, we query $R(a, x)$ and $R(b, x)$ for all $x \in V$, and check if $|R(a, x) - R(b, x)| = 1$ holds for all of them. Therefore, we obtain the following algorithm.

Theorem 14 *Let $G = (V, E)$ be an undirected, weighted or unweighted graph, and let $n = |V|$, and let $a, b \in V$. There exists an algorithm that decides whether (a, b) is a cut edge using $2n - 3$ ER queries.*

4. Property testing

Next, we present our algorithms for testing properties of a hidden graph using ER queries. A *graph property* is a set of graphs that is closed under isomorphism, such as connectivity, acyclicity or planarity. The algorithms in this section are approximate in the sense that they guarantee correctness only if the input graph either possesses the desired property or is far from having it. This approximate testing model has been extensively studied within the property testing framework Goldreich (2017). Our results are most closely related to the *bounded-degree model*.

In this model the input graph is assumed to have maximum degree d, which is bounded by a constant. The algorithm can query the neighbors of a vertex. A graph is ε-far from having a property if more than εdn edge modifications are required to make it satisfy that property. The query complexity in this models is often described as a function of n and ε. See Goldreich (2017, Section 9) for more on this model.

In this section, we provide property testing algorithms for vertex biconnectivity and edge biconnectivity. Additionally, we present a theorem that allows us to adapt existing property testing algorithms to our query model. The number of queries required depends on the *ER density*, a graph parameter that we introduce and define. We demonstrate that bounded-degree, bounded-treewidth graphs have small ER density, making them well-suited for efficient property testing algorithms under our framework.

4.1. Testing vertex biconnectivity

First, we use Theorem 12 to show that the vertex biconnectivity of a graph can be tested with a linear number of queries.

Theorem 15 *Let $\varepsilon > 0$ and let $G = (V, E)$ be an undirected graph with n vertices. There exists an algorithm that makes $O(n/\varepsilon)$ effective resistance queries that accepts with probability 1 if G is vertex biconnected, and rejects with probability at least $2/3$ if G is ε-far from being vertex biconnected, i.e., one needs to add at least εm edges to make G vertex biconnected.*

Proof First, we check that G is connected by making $n - 1$ queries. Specifically, we check that an arbitrary vertex v is at finite distance from all other $n - 1$ vertices. If G is not connected we reject. So, we assume G is connected in the rest of the proof, which in particular implies that $m \geq n - 1$.

Our algorithm takes an arbitrary vertex $v \in V$, and samples $s = 4/\varepsilon$ other vertices $u_1, \ldots, u_s$ uniformly at random. It rejects if any of the following conditions holds: (1) v is a cut vertex, or (2) there exists $1 \leq i \leq s$ such that v and u_i belong to different biconnected components. If none of these two conditions hold, our algorithm accepts. Both of these conditions can be checked with $O(s \cdot n) = O(n/\varepsilon)$ ER queries using the algorithm of Theorem 12.

If G is biconnected then none of the two conditions hold, thus our algorithm accepts.

Now, suppose G is ε-far from being biconnected. If v is a cut vertex our algorithm rejects, so suppose v is not a cut vertex, and it blongs to a biconnected component B.

As G is ε-far from being biconnected, it contains at least $\varepsilon m + 1 \geq \varepsilon(n - 1) + 1$ biconnected components. Therefore, its largest biconnected component, and in particular B, has at most $n - \varepsilon(n - 1) = n - \varepsilon n + \varepsilon \leq n - \varepsilon n + \varepsilon(n/2) = (1 - \varepsilon/2)n$ vertices. Therefore, with probability at least $1 - (1 - \varepsilon/2)^s \geq 1 - e^{-(\varepsilon/2) \cdot s} \geq 1 - e^{-2} \geq 2/3$, there exists a $1 \leq i \leq s$ such that $u_i \notin B$. Since, v is not a cut vertex, in this case, v and u_i belong to two different biconnected components, hence our algorithm rejects. ∎

4.2. Local reconstruction via Schur complement

Our result for adapting known property testing algorithms relies on an algorithm that identifies the neighbors of a given vertex with ER queries. This algorithm exploits properties of the Schur complement stated in Theorem 6, as well as the algorithm of Theorem 1 for full reconstruction of a graph from its pairwise ER distances.

Corollary 16 *Let $G = (V, E)$ be a weighted graph, and let $U \subseteq V$. The Schur complement of G on U, G_U, can be computed with $\binom{|U|}{2}$ effective resistance queries.*

We denote the *unit ball* around a vertex $v \in V$ with respect to the effective resistance metric by $B_R(v) := \{u \in V \mid R(u, v) \le 1\}$. It follows from Theorem 16 that we can find neighbors of a given vertex, provided that there are not too many vertices in its unit ball in the ER metric of the graph.

Corollary 17 *Let $G = (V, E)$ be an unweighted graph and $u \in V$. Then, one can find all neighbors of u with $O(n + |B_R(u)|^2)$ ER queries.*

Proof We start by computing $B_R(u)$, which we can do by querying for the ER distance between u and each of the other $n - 1$ vertices. All neighbors of u in G are at ER distance at most one by Theorem 4, and hence contained in $B_R(u)$. Therefore, by Theorem 6, u's neighborhood is preserved in the Schur complement $G_{B_R(u)}$. So, we can find the neighbors of u by constructing the Schur complement, $G_{B_R(u)}$, which can be done with $O(|B_R(u)|^2)$ ER queries by Theorem 16. ∎

We will use Theorem 17 in two different ways: to test edge biconnectivity for a general graph, and to adapt property testing algorithms for graphs whose unit balls contain a bounded number of vertices.

4.3. Testing edge biconnectivity

Next, we show an algorithm to test edge biconnectivity of a graph. Our algorithm relies on Theorem 17 to find neighbors of vertices, which facilitates breadth-first search in a hidden graph.

Theorem 18 *Let $G = (V, E)$ be an undirected graph with n vertices and let $0 < \varepsilon \le 1$. There exists an algorithm that makes $O(n/\varepsilon^2 + 1/\varepsilon^4)$ ER queries and behaves as follows. It accepts with probability 1 if G is edge biconnected, and rejects with probability $\ge 2/3$ if G is ε-far from being edge biconnected, i.e., if one needs to add more than $\varepsilon \cdot m$ edges to make G edge biconnected.*

Proof As in the proof of Theorem 15, we first check that G is connected using $n - 1$ ER queries, and if it is not we reject. So, we assume G is connected in the rest of the proof, which in particular implies that $m \ge n - 1$.

Our algorithm samples a set $S \subseteq V$ of $16/\varepsilon$ vertices uniformly at random. We call a biconnected component *small* if its size is at most $4/\varepsilon$. We call a biconnected component *low-degree* if it is incident to at most two cut edges. A tree can be obtained by contracting all edge-biconnected components of the graph. Since the average degree of any tree is less than two, it follows that at least half of the biconnected components must have a low degree.

At a high level, our algorithm checks if any of the sampled vertices in S belong to a biconnected component that is both small and low-degree with a BFS-like algorithm. It rejects if it finds such a biconnected component and accepts otherwise. For each $s \in S$, our algorithm simulates a BFS starting at s, that can end in three different ways: (1) the BFS finds a cut edge, hence it decides that G is not edge biconnected, (2) the BFS finds more than $4/\varepsilon + 2$ vertices that are either in the same biconnected component of s, or are incident to cut edges of this biconnected component, hence it decides that the biconnected component of s is not small or not low-degree.

To simulate BFS, upon taking a vertex u from the queue, we query all effective resistances from u, and compute the unit ER ball centered at u, $B_R(u)$. Note, every vertex that is strictly inside this ball is in the same edge biconnected component of u. Also, every vertex on the surface of this ball, is either in the same edge biconnected component, or it is adjacent to u via a cut edge (Theorem 4).

Therefore, if $|B_R(u)| > 4/\varepsilon + 2$, the edge biconnected component of u (and also the sampled point that is the root of this BFS call) cannot be both small and low-degree. In this case, we continue to the next sample point. Otherwise, if $|B_R(u)| \leq 4/\varepsilon + 2$, we compute the neighbors of u with another $|B_R(u)|^2 = O(1/\varepsilon^2)$ queries, using the algorithm of Theorem 17, and check if any of the incident edges of u is a cut edge, by checking if its effective resistance is exactly one, using Theorem 4. If we find a cut edge then we reject. Otherwise, we continue the BFS until we visit at least $4/\varepsilon$ vertices. In that case, we continue to the next sampled point, as the biconnected component of the current sampled point is larger than $4/\varepsilon$, and hence not small.

For each vertex v that the BFS visits, it spends $O(n)$ queries to compute $B_R(v)$, and $O(1/\varepsilon^2)$ queries to compute the neighbors of v. For each sampled vertex in S, BFS visits $O(1/\varepsilon)$ vertices, the size limit for low-degree small biconnected components. Therefore, the total number of queries is $|S| \cdot O(1/\varepsilon) \cdot O(n + 1/\varepsilon^2) = O(n/\varepsilon^2 + 1/\varepsilon^4)$.

If G is edge biconnected, our algorithm accepts with probability one, as it will not find a cut edge.

If G is ε-far from edge biconnectivity, then it contains at least $\varepsilon m + 1 \geq \varepsilon(n-1)$ edge biconnected components. At least half of them, i.e. $\varepsilon(n-1)/2$ of them, are low-degree. On the other hand, G (which has n vertices) can contain at most $\varepsilon n/4$ edge biconnected components of size at least $4/\varepsilon$. Thus, assuming $\varepsilon \leq 1$ and $n \geq 8$, there are at least $\varepsilon(n-1)/2 - \varepsilon n/4 \geq \varepsilon n/8$ small edge biconnected components, i.e., of size at most $4/\varepsilon$, that are incident to at most two cut edges. In particular, these low-degree small biconnected components have at least $\varepsilon n/8$ vertices in total. Hence, our sample S (of size $16/\varepsilon$) contains at least one vertex from a low-degree small biconnected component with probability at least $1 - (1 - \varepsilon/8)^{-16/\varepsilon} \geq 1 - e^{-2} \geq 2/3$. Therefore, our algorithm rejects with at least this probability. ∎

4.4. The ER density and property testing with ER queries

Theorem 17 implies the following meta theorem, which allows us to adapt any property testing algorithm within the bounded-degree model to a property testing algorithm with ER queries. The query complexity of our algorithm depends on the ER density $\rho_R(G)$ of the graph G, which we define as the maximum size of any ER unit ball, i.e., $\rho_R(G) := \max_{v \in V} |B_R(v)|$.

Theorem 19 *Let $G = (V, E)$ be a graph with ER density $\rho = \rho_R(G)$. Let A be any property testing algorithm for problem P in the bounded-degree model with $f(\varepsilon, n)$ running time. Then, there is a property testing algorithm for problem P that uses $O(f(\varepsilon, n) \cdot (n + \rho^2))$ effective resistance queries.*

Remark 20 *We remark that the linear dependency on n in Theorem 18 and Theorem 19 can be eliminated if we have an oracle that returns the vertices in the order of their distance to a vertex v, allowing us to stop querying at any point. Then, to discover the vertices in a unit ball around v, $B_R(v)$, we only need to spend $|B_R(v)|$ queries as opposed to $n - 1$.*

Property testing in the bounded-degree model is a well-studied subject with many fundamental results. To name a few, there are $f(1/\varepsilon)$ time testers to decide if a graph (1) contains a fixed subgraph, (2) is regular, (3) is k-connected, or (4) belongs to a minor closed family (e.g., planar graphs). (See Goldreich (2017) for a more extensive list of results.)

Theorem 19 can be used to adapt all these results to algorithms with ER queries. However, this comes with an additional cost of a factor of $(n + \rho^2)$. There exist graphs for which ρ^2 is large, making the theorem less useful—for example, in the case of expander graphs.

However, we show that ρ is bounded for bounded-degree bounded treewidth graphs. As a result, all the aforementioned results from the bounded-degree model can be adapted to bounded treewidth graphs with an additional factor of n in the number of queries. In most cases, this leads to a linear number of ER queries, as opposed to the quadratic number required when reconstructing the entire graph.

ER density of bounded treewidth graphs. We use the following two fundamental results of Bodlaender (1999) on domino tree decompostion and Nash-Williams (1959) on a lower bound for ER distance of two vertices towards showing that bounded-degree bounded treewidth graphs have boudned ER density.

Lemma 21 (Bodlaender (1999, Theorem 3.1)) *Let $G = (V, E)$ be a graph with maximim degree d_G and treewidth w_G. Then, G has a tree decomposition of width $O(w_G d_G^2)$, in which each vertex is in at most two bags.* [3] *Moreover, the degree of each bag in the tree decomposition is $O(w_G d_G^2)$.*

The last part of the statement is not explicit in Bodlaender (1999). However, it is easy to conclude by observing that each neighbor of a bag B has at least one vertex in common with B, assuming the underlying graph is connected. Therefore, since each vertex appears in at most two bags, the number of neighbors of a bag cannot be larger than its size.

Lemma 22 (Nash-Williams (1959)) *Let $G = (V, E)$ be a graph and let $s, t \in V$. Let $S_1, S_2, \ldots, S_k$ be such that $s \in S_i$ and $t \notin S_i$ for all $i \in [k]$, and such that the edge sets $E(S_i, V \setminus S_i)$ are pairwise disjoint. Then,*

$$R(s,t) \geq \sum_{i=1}^{k} \frac{1}{|E(S_i, V \setminus S_i)|} \, .$$

Lemma 23 *Let $G = (V, E)$ be a graph with maximum degree d_G, and let $(\mathcal{B}, T)$ be a tree decomposition of G with width w_T and maximum degree d_T. Also, suppose that each vertex of G appears in at most b_T bags of T. Let $s, t \in V$, let $B_s, B_t \in \mathcal{B}$ be any two bags that contain s and t, respectively, and let $r_T(s, t)$ be the distance of these bags in T.*

$$r_T(s,t) \leq 4b_T \cdot d_G \cdot w_T \cdot R(s,t) \, .$$

Proof If $r_T(s, t) < 4b_T$, the statement of the lemma holds. Thus, assume $r_T(s, t) \geq 4b_T$.

Let $p = r_T(s, t)$, and let $B_s = B_1, \ldots, B_p = B_t$ be the unique B_s, B_t path in T. We know that each vertex of G is in at most b_T bags. Moreover, we know that these bags form a connected induced tree in the tree decomposition. Therefore, for each $i \in [p - b_T]$, B_i and B_{i+b_T+1} are disjoint. We conclude that there is a subsequence $B'_1, \ldots, B'_q$ of $B_1, \ldots, B_p$ such that (1) $q \geq \lfloor p/b_T \rfloor - 2$, (2) $s, t \notin B'_i$ for $i \in [q]$, and (3) for each $1 \leq i < j \leq q$, B'_i and B'_j are disjoint (we need -2 in condition (1) because of condition (2)). Note, $q \geq 1$ because $p \geq 4b_T$.

---

3. Such a decomposition is called a *domino decomposition*, and the smallest width of a domino decomposition is called *domino treewidth*.

Note, for each $i \in [q]$, B_i' is a vertex cut that separates s and t in G. Let $S_1^i, \ldots, S_{r_i}^i$ be the connected components of $V \setminus B_i'$ with $s \in S_1^i$ and $t \in S_{r_i}^i$. Then, let $S_i' := S_1^i \cup \cdots \cup S_{r_i-1}^i \cup B_i'$, and observe that $s \in S_i'$ and $t \notin S_i'$.

Moreover, for each $i \in [q]$, each edge in $E(S_i', V \setminus S_i')$ has one endpoint in B_i' (note, in particular, $V \setminus S_i' = S_{r_i}^i$). Since the bags B_i' are disjoint, we conclude that the edge sets $E(S_i', V \setminus S_i')$ for $i \in [q]$ are disjoint. We also conclude that for every $i \in [q]$,

$$|E(S_i', V \setminus S_i')| \leq d_G \cdot |B_i'| \leq d_G \cdot w_T.$$

It follows by Theorem 22 that

$$R(s,t) \geq \frac{q}{d_G \cdot w_T} \geq \frac{\lfloor p/b_T \rfloor - 2}{d_G \cdot w_T} \geq \frac{p/b_T - 3}{d_G \cdot w_T}$$
$$\geq \frac{p/b_T - (3/4) \cdot (p/b_T)}{d_G \cdot w_T} = \frac{(1/4) \cdot p}{b_T \cdot d_G \cdot w_T} = \frac{(1/4) \cdot r_T(s,t)}{b_T \cdot d_G \cdot w_T},$$

where the last inequality holds as $p \geq 4b_T \Rightarrow 3 \leq (3/4) \cdot (p/b_T)$. ∎

Theorem 24 *Let $G = (V, E)$ be a graph with maximum degree d_G and treewidth w_G. Then the ER density $\rho_R(G)$ of G is $(w_G d_G)^{O(w_G d_G^3)}$. In particular, a graph with constant maximum degree and constant treewidth has constant ER density.*

Proof By Theorem 21 there exists a tree decomposition T of G with maximum degree $d_T = O(w_G d_G^2)$ and width $w_T = O(w_G d_G^2)$ in which each vertex appears in at most $b_T = 2$ bags.

Let $s \in V$, and let B_s be h the bag in T that contains s. By Theorem 23, vertices with ER distance at most one from s (i.e., vertices in $B_R(s)$), are in bags of distance at most $\alpha := \max(4b_T \cdot d_G \cdot w_T, 4b_T) = 4b_T \cdot d_G \cdot w_T$ from B_s in T.

There are at most $\sum_{i=0}^{\alpha} d_T^i = (d_T^{\alpha+1} - 1)/(d_T - 1) \leq d_T^{\alpha+1}$ such bags, assuming $d_T \geq 2$, that collectively contain at most

$$w_T \cdot d_T^{\alpha} = w_T \cdot d_T^{4b_T \cdot d_G \cdot w_T + 1} = w_G d_G^2 \cdot (w_G d_G^2)^{O(d_G^3 \cdot w_G)} = (w_G d_G)^{O(d_G^3 \cdot w_G)}$$

vertices. ∎

Therefore, the ER density of a graph G is a constant if it has constant maximum degree and constant treewidth. In that case, Theorem 19 implies a property testing algorithm with a linear number of ER queries, if a constant-time property testing algorithm is known in the bounded-degree model.

ER density of bounded-degree planar graphs. We next show that bounded-degree planar graphs are another class of graphs that have bounded ER-density. This follows from the following lemma in Jonasson and Schramm's work on covering times on planar graphs Jonasson and Schramm (2000).

Lemma 25 ((Jonasson and Schramm, 2000, Lemma 3.1)) *Let $d \in \mathbb{Z}^+$ be a constant. Then there exist positive real constants $c_d, r_d \in \mathbb{R}$ such that the following holds for every planar graph $G = (V, E)$ with maximum degree d. For every subset of vertices $W \subseteq V$, there exists $V' \subseteq W$ with $|V'| \geq |W|^{c_d}$ such that for any distinct $u, v \in V'$, $R(u, v) \geq r_d \cdot \log |W|$.*

We next derive the fact that bounded-degree planar graphs have bounded ER-density as a corollary of Theorem 25.

Corollary 26 *Let $G = (V, E)$ be a planar graph with maximum degree d. Then the ER density $\rho_R(G)$ of G is $O_d(1)$, i.e., at most a constant depending only on d.*

Proof Let $v \in V$ be such that $|B_R(v)| = \rho_R(G)$ where $B_R(v) := \{u \in V \mid R(u,v) \leq 1\}$. Then by Theorem 25, there exist constants $c_d, r_d > 0$ and a subset $V' \subseteq B_R(v), |V'| \geq (\rho_R(G))^{c_d}$ such that for all distinct $u, w \in V'$, $R(u,w) \geq r_d \cdot \log(\rho_R(G))$. In particular, there exist $u, w \in B_R(v)$ such that $R(u,w) \geq r_d \cdot \log(\rho_R(G))$ since if $|V'| \leq 1$ then $\rho_R(G)^{c_d} \leq 1$ and the argument would be finished. However, the assumption that $u, w \in B_R(v)$ and the triangle inequality imply that $R(u,w) \leq 2$, and so $r_d \cdot \log(\rho_R(G)) \leq 2$. It follows that $\rho_R(G) \leq 2^{2/r_d} = O_d(1)$, as needed. ∎

5. Shortest Path versus Effective Resistance Queries

In this section, we motivate the study of effective resistance queries by showing that they are incomparable in power to shortest path queries. Specifically, we show that sometimes it is (provably) possible to use fewer effective resistance queries to solve a given task, and sometimes it is possible to use fewer shortest path queries.

Our first result asserts that it is more efficient to check whether a graph is a clique using effective resistance queries than shortest path queries.

Theorem 27 *There is an algorithm for testing whether a graph G with n vertices is a clique (i.e., whether $G = K_n$) using $O(n)$ effective resistance queries, but any such algorithm requires $\Omega(n^2)$ shortest path queries.*

Proof As shown in Theorem 11, it is possible to check using $O(n)$ effective resistance queries whether two graphs G and H are equal, assuming that G is a subgraph of H. The algorithm for checking that G is a clique follows from the special case of this result when $H = K_n$, as every graph is a subgraph of K_n.

The lower bound for shortest path queries works as follows. Suppose that G is either equal to K_n or equal to $K_n \setminus \{e\}$ for a single unknown edge e. Any shortest path query on u, v will return the same value (i.e., 1) in either of these two cases, except when u and v are the endpoints of e. It follows that any algorithm requires $\binom{n}{2}$ shortest path queries. ∎

Our second result asserts that it is more efficient to check whether two given vertices in G are adjacent using shortest path queries than effective resistance queries.

Theorem 28 *There is an algorithm for testing whether two given vertices $u, v \in V$ in a simple graph are adjacent using 1 shortest path query, but any algorithm for this problem requires $\Omega(n)$ effective resistance queries.*

Proof It is possible to check whether u, v are adjacent using a single shortest path query since u and v are adjacent if and only if the length of the shortest path between them is 1.

On the other hand, consider the problem of deciding whether the underlying graph is (1) G or (2) of the form $H = H_{i,j}$ for some i, j using effective resistance queries, where these graphs

Figure 1: A pair of unweighted graphs G and $H_{i,j}$ on n vertices that require $\Omega(n)$ effective resistance queries to distinguish; see Theorem 28. The effective resistance between v_1, v_2 is the same in both graphs (i.e., $R_G(v_1, v_2) = R_H(v_1, v_2) = 1$) despite v_1, v_2 being adjacent in G but not in $H_{i,j}$.

are as shown in Figure 1. I.e., G consists of two star graphs on $n/2$ vertices whose centers v_1, v_2 are connected by an edge, and $H = H_{i,j}$ consists of two star graphs on $n/2 - 1$ vertices whose centers v_1, v_2 form a square with the vertices v_i, v_j, where $i, j \in \{3, \ldots, n\}$, $i \neq j$ are unknown. In particular, the graphs G and H are on the same set of vertices $V = \{v_1, \ldots, v_n\}$, but in G, v_1 and v_2 are adjacent, and in H, v_1 and v_2 are not adjacent. So, the problem of deciding whether the underlying graph is G or H reduces to checking whether v_1 and v_2 are adjacent. To show that checking adjacency requires $\Omega(n)$ effective resistance queries, it therefore suffices to show that determining whether the underlying graph is G or H requires $\Omega(n)$ such queries.

One can check that $R_G(x, y) = R_H(x, y)$ for all $x, y \in V \setminus \{v_i, v_j\}$, and in particular that $R_G(v_1, v_2) = R_H(v_1, v_2) = 1$. So, to decide whether the underlying graph is G or H, it is necessary to perform a query involving either v_i or v_j. However, i, j are unknown, and so any (possibly adaptive) sequence of queries $R(x, y)$ necessarily including v_i and v_j must be of length at least $(n - 2)/2 = \Omega(n)$. ∎

Put together, Theorems 27 and 28 show that shortest path queries and effective resistance queries can each be asymptotically more efficient than the other for some verification tasks, and so neither type of query subsumes the other—they are incomparable. Understanding their relative power is therefore quite interesting. We also note that Theorem 28 shows that testing whether a given pair of vertices is adjacent takes $\Omega(n)$ effective resistance queries, but we do not know if this is tight. In fact, we do not have a better upper bound than the trivial $\binom{n}{2}$ queries needed to recover the whole graph. Finding tight bounds for this problem is therefore an interesting question.

6. Graph Reconstruction via Effective Resistance Queries

We give algorithms solving two graph inference problems, the first being full reconstruction of a graph from resistance distance queries given a tree decomposition, and the second being the problem of completing a graph for which the adjacency matrix is only partially known. For the former, , we utilize the technique of reconstructing the Schur complement for a subset of vertices to obtain the overall graph. For the latter one, we rely on the property that effective resistance appears as directional derivatives of certain functions of the space of positive definite matrices.

6.1. Graph Reconstruction from a Tree Decomposition

Next, we show that a graph can be reconstructed quickly provides a tree decomposition of a small width. Estimating such a tree decomposition is a more challenging task that we leave as an open question. Our results in this section relies on properties of Schur complement of Theorem 6.

Corollary 29 *Let $G = (V, E)$ be a graph, and let $(\mathcal{B}, T)$ be a tree decomposition of G of width $k \in \mathbb{N}$. Also, let $B \subseteq \mathcal{B}$ be a bag of T that has degree one, and let $P \subseteq \mathcal{B}$ be the parent of B. For all vertices $u \in B \setminus P$ we can determine their neighborhood using $\binom{k}{2}$ effective resistance queries.*

Proof Any vertex $u \in B \setminus P$ does not appear in any other bag of T, because vertices only appear in connected induced trees of T. Thus, all its neighbors have to appear in B, because every edge of the graph must appear in at least one bag. It follows, by Theorem 6 that the neighborhood of u is identical in G and the Schur complement $G_{B \setminus P}$. Thus, we can discover them by reconstructing $G_{B \setminus P}$, with $\binom{|B|}{2} = \binom{k}{2}$ ER queries using Theorem 1. ∎

Using this, we can obtain the adjacency of vertices that belong only in a leaf bag. Recursively, the vertices exclusive to that leaf bag can be eliminated via the Schur complement. What we show is that removing this bag, as a node, from the tree decomposition, gives a valid tree decomposition for the new graph, with those vertices being eliminated.

Lemma 30 *Let $G = (V, E)$ be a graph with a tree decomposition $(\mathcal{B}, T)$ of treewidth k. For any bag of degree one, $B \in \mathcal{B}$, and its parent $P \in \mathcal{B}$, the subtree T' induced by $\mathcal{B}' = \mathcal{B} \setminus \{B\}$ forms a valid tree decomposition of the Schur complement graph $G' = G_{V \setminus (B \setminus P)}$.*

Proof We must show (1) every vertex of $V \setminus (B \setminus P)$ is present in a bag of $\mathcal{B}'$, (2) for any edge in G' there is a bag in $\mathcal{B}'$ that contains both of its endpoints, and (3) every vertex of of G' exists in a contiguous subtree of T'. Property (3) is guaranteed since we removed a leaf from T to obtain T', which does not disconnect any induced subtree of T.

To show property (1), note the vertex set of G' is exactly $V \setminus (B \setminus P)$. Let $v \in V \setminus (B \setminus P)$. If $v \notin B$ then there is a bag $B' \neq B$ such that $v \in B'$, otherwise $(\mathcal{B}, T)$ is not a valid tree decomposition. Otherwise, if $v \in B$ then it must be the case that $v \in P$, since $v \in V \setminus (B \setminus P)$. Therefore, v is in a bag of $\mathcal{B}'$, namely P.

Finally, we show property (3). For any edge e in the Schur complement graph $G_{V \setminus (B \setminus P)}$, either $e \in E$ or e is created during the Schur complement process. If $e \in E$ then it is in a bag of $\mathcal{B}$, therefore, in a bag of $\mathcal{B} \setminus \{B\}$. Otherwise, $e = (u, v)$ such that u, v are in the neighborhood of $B \setminus P$, as the edge is created after reducing $B \setminus P$. In this case, since all the neighbors of $B \setminus P$ are in P, we have that $u, v \in P$, as desired. ∎

Having shown both that we can obtain the entire adjacency of 'interior vertices' of a leaf bag, and that removing such bags gives a tree decomposition of the Schur complement with those vertices eliminated, the algorithm becomes clear.

Theorem 31 *Let $G = (V, E)$ be an undirected weighted graph and suppose that a tree decomposition $(\mathcal{B}, T)$ of G is given that has width k. There exists an algorithm to reconstruct G in $O(k^2 n)$ ER queries.*

Proof Our algorithm inductively reconstructs G by considering its tree decomposition T. Let B be a bag of T with exactly one neighbor P in T, and let $U = B \backslash P$. We can assume U is non-empty, as otherwise, we can drop B and obtain a tree decomposition of T with fewer bags. We can compute all the neighbors of all the vertices in U with $O(k^2)$ ER queries by Theorem 29. Then, we recursively compute the Schur complement $G_{V \backslash U}$. To that end, we use the fact that $T \backslash \{B\}$ is a valid tree decomposition of $G_{V \backslash U}$, which holds by Theorem 30, and that the ER queries between two vertices of $V \backslash U$ return their effective resistance in this Schur complement (which is equal to their effective resistance in G).

To finish our reconstruction, we need to build G provided the neighborhood of U (computed by the algorithm of Theorem 29) and the Schur complement $G_{V \backslash U}$ (computed recursively). To that end, let L be the Laplacian of G, and observe that we know $L(U, \overline{U})$, $L(\overline{U}, U)$ and $L(U, U)$, since we know the neighbors of every vertex in U. Moreover, we know $L_{\overline{U}}$ from our recursive computation. Hence, we can compute $L(\overline{U}, \overline{U})$, as by the definition of Schur complement (Equation (4)),

$$L(\overline{U}, \overline{U}) = L_{\overline{U}} + L(\overline{U}, U)L(U, U)^+ L(U, \overline{U}).$$

Overall, we know all four submatrices of L, hence we know L. ■

6.2. Graph Completion with Minimal Query Complexity

Next, we consider the problem of graph completion. Let $G = (V, E)$ be a connected weighted graph whose adjacency matrix is partially known, with exactly k unknown entries. The goal of the graph completion problem is to identify these unknown entries using the minimum number of ER queries.

It is straightforward to recover G using $O(n^2)$ ER queries by simply ignoring the known part of the adjacency matrix and applying Theorem 1. Furthermore, by leveraging the Schur complement, one can recover G with $O(k^2)$ ER queries (Theorem 32). However, we show that k ER queries are sufficient to reconstruct G when the edge weights are drawn from a finite set, and in particular, for unweighted graphs (Theorem 37).

6.2.1. Quadratically Many ER Queries

To warm up, we present the following simple application of the Schur complement together with the known algorithm for fully reconstructing a graph, to recover G with $O(k^2)$ ER queries.

Theorem 32 *Let $G = (V, E, w_G)$ be an weighted graph. Suppose k entries of the adjacency matrix of G are not known. There exists an algorithm to reconstruct G (i.e. to find the missing k weights) with $O(k^2)$ ER queries. (The running time of the algorithm is $\mathrm{poly}(k) \cdot n^2$).*

Proof Let L be the Laplacian of G. Let U be the set of vertices that are incident to the edges with unknown weights, and note $|U| \leq 2k$. We use $\binom{2k}{2}$ ER queries to obtain all ER values between all pairs of vertices of U, and use Theorem 1 to compute the Schur complement L_U. Then, since $L(U, \overline{U})$, $L(\overline{U}, U)$ and $L(\overline{U}, \overline{U})$ are already known, we can compute $L(U, U)$ from the Schur complement formula, Equation (4), $L_U = L(U, U) - L(U, \overline{U})L(\overline{U}, \overline{U})^+ L(\overline{U}, U)$. Therefore, we can have all four submatrices of L, hence G is recovered. ■

6.2.2. LINEARLY MANY ER QUERIES

Next, we show that the effective resistance values between every pair of vertices whose corresponding adjacency matrix entry is missing are sufficient to reconstruct G. We show that these ER values, together with the known adjacency values, uniquely determine the graph. Specifically, given an adjacency matrix of G with k entries missing we are able to fully reconstruct the adjacency matrix using k ER queries.

The proof of this uniqueness result is, at a high level, very similar to the idea presented in the MathOverflow post by Speyer (2021). However, we needed to adapt that idea to our setting with effective resistance (Theorem 36) and establish a connection between the derivative of the function studied there and effective resistance (Theorem 35). The application of this work to the context of effective resistance is not immediately obvious, as, firstly, the Laplacian is not an invertible matrix, and secondly, the construction of a function with strict convexity whose gradient is equal to a certain set of effective resistance is non-trivial.

Basic matrix calculus. We briefly introduce some background from matrix calculus that we use in this section. We refer the reader to Magnus and Neudecker (2019) for a comprehensive review.

We use $X = (x_{i,j})_{(i,j)\in[n]\times[n]}$ to denote an $n \times n$ matrix of variables $x_{i,j}$. We work with functions $f : \mathbb{R}^{n\times n} \to \mathbb{R}$ that map $n \times n$ matrices to real numbers. The *derivative* of such a function f is an $n \times n$ matrix, denoted

$$\frac{\partial f}{\partial X} = \left(\frac{\partial f}{\partial x_{i,j}}\right)_{(i,j)\in[n]\times[n]}.$$

A differentiable function $f : D \to \mathbb{R}$ for $D \subseteq \mathbb{R}^m$ is *strictly concave on D* if D is convex and if for any $\boldsymbol{a}, \boldsymbol{b} \in D$,

$$f(\boldsymbol{a}) - f(\boldsymbol{b}) < \left(\frac{\partial f(\boldsymbol{a})}{\partial \boldsymbol{x}}\right)^T (\boldsymbol{b} - \boldsymbol{a}).$$

See Boyd and Vandenberghe (2004, Section 3.1.3).

One can show that strict concavity implies uniqueness of gradients over the domain, i.e., that $\partial f(\boldsymbol{a})/\partial \boldsymbol{x} \neq \partial f(\boldsymbol{b})/\partial \boldsymbol{x}$ for $\boldsymbol{a}, \boldsymbol{b} \in D, \boldsymbol{a} \neq \boldsymbol{b}$. Hence, we have the following lemma.

Lemma 33 ((Griewank et al., 1991, Theorem 1)) *Let $f : D \to \mathbb{R}$ be a strictly concave function on a convex domain $D \subseteq \mathbb{R}^m$. Then $\partial f(\boldsymbol{a})/\partial \boldsymbol{x} \neq \partial f(\boldsymbol{b})/\partial \boldsymbol{x}$ for any distinct $\boldsymbol{a}, \boldsymbol{b} \in D$.*

We consider the function $\log\det(\cdot)$ defined on the set of $n\times n$ matrices. We specifically work on the restriction of this function on the set of (symmetric) positive definite matrices, denoted $S_n^{++} \subset \mathbb{R}^{n\times n}$. Positive definite matrices are invertible since all their eigenvalues are non-zero. Furthermore, the set S_n^{++} is known to form a cone and, in particular, is convex. We use the following fact, a proof of which can be found in Boyd and Vandenberghe (2004, Section 3.5.1).

Lemma 34 *The function $f : \mathbb{R}^{n\times n} \to \mathbb{R}$ be the function defined as $f(X) = \log\det(X)$ is strictly concave on the set of positive definite matrices.*

Uniqueness and reconstruction. We are now ready to present the main result. First, we establish the uniqueness of graph completion given the ER values corresponding to the pairs of vertices with unknown entries in the adjacency matrix.

To this end, we study the function $f = \log\det(\cdot)$ and its derivative on the space of positive definite matrices, which includes regularized Laplacians (recall Equation (3)). First, we show that the directional derivatives of f in certain directions correspond to the effective resistances in graphs. In the following proof, we define $E_{s,t}$ to be the $n \times n$ matrix whose (s, t) entry is equal to one and has all other entries equal to zero. Recall from Section 2 that the *edge Laplacian* of an edge $(s, t) \in E$ is the matrix $L_{s,t} := E_{s,s} + E_{t,t} - E_{s,t} - E_{t,s}$. It is called the edge Laplacian as it is Laplacian of the graph with n vertices and one edge (i, j).

In this section, we will view the regularized Laplacian of a graph as a function of its edge weights. Specifically, define the map $\tilde{L} : \mathbb{R}_{\geq 0}^{\binom{n}{2}} \to \mathbb{R}^{n \times n}$ as $\tilde{L}(\boldsymbol{w}) = \left(\sum_{s,t \in \binom{V}{2}} w_{s,t} L_{s,t} \right) + \frac{1}{n} J$, i.e, the regularized Laplacian of the graph with weights $\boldsymbol{w}$. Likewise, denote the graph with weights $\boldsymbol{w}$ as $G(\boldsymbol{w})$. A pair of nodes with weights $w_{s,t} = 0$ means the edge (s, t) is not present in the graph.

Lemma 35 *Let $\boldsymbol{w} \in \mathbb{R}_{\geq 0}^{\binom{n}{2}}$ be a set of graph weights such that $G(\boldsymbol{w})$ is connected. Then for any $s, t \in V$, we have*

$$\frac{\partial \log\det \tilde{L}(\boldsymbol{w})}{\partial w_{s,t}} = R_{G(\boldsymbol{w})}(s, t) \,.$$

Proof Using the chain rule,

$$\frac{\partial \log\det(\tilde{L}(\boldsymbol{w}))}{\partial w_{s,t}} = \frac{\partial \log\det(\tilde{L}(\boldsymbol{w}))}{\partial \tilde{L}(\boldsymbol{w})} \cdot \frac{\partial \tilde{L}(\boldsymbol{w})}{\partial w_{s,t}} \,.$$

By (Petersen and Pedersen, 2012, Equation 51), $\frac{\partial \log\det \tilde{L}(\boldsymbol{w})}{\partial \tilde{L}(\boldsymbol{w})} = \tilde{L}(\boldsymbol{w})^{-1}$. The partial derivative $\frac{\partial \tilde{L}(\boldsymbol{w})}{\partial w_{s,t}} = L_{s,t}$. Therefore,

$$\frac{\partial \log\det \tilde{L}(\boldsymbol{w})}{\partial \tilde{L}(\boldsymbol{w})} \cdot \frac{\partial \tilde{L}(\boldsymbol{w})}{\partial w_{st}} = \tilde{L}^{-1}(\boldsymbol{w}) \cdot L_{s,t} = \tilde{L}_{s,s}^{-1}(\boldsymbol{w}) + \tilde{L}_{t,t}^{-1}(\boldsymbol{w}) - \tilde{L}_{s,t}^{-1}(\boldsymbol{w}) - \tilde{L}_{t,s}^{-1}(\boldsymbol{w}) = R_{G(\boldsymbol{w})}(s, t) \,,$$

where $\cdot$ is the inner product of matrices, and the second equality follows from Theorem 5, Item 2. $\blacksquare$

Next, we present our uniqueness result based on the lemma above. At a high level, we consider the space W of all regularized Laplacians that are consistent with the known part of G. We show that the function $\log\det(\cdot)$ has distinct derivatives at every pair of points in W. Moreover, these derivatives are uniquely determined by the effective resistances between pairs of vertices with unknown adjacency entries. We conclude that knowing ER values between pairs of vertices with missing adjacency values uniquely determines G.

Let $G = (V, E, w)$ be a connected weighted graph with n vertices and non-negative edge weights. Let $K \sqcup U$ be a partition of $\binom{V}{2}$ such that the weights on K are known and the weights of U are unknown. Define the function $\tilde{L}_K : \mathbb{R}^U \to \mathbb{R}^{n \times n}$ as

$$\tilde{L}_K(\boldsymbol{x}) := \sum_{(u,v) \in K} w_{u,v} L_{u,v} + \sum_{(s,t) \in U} x_{s,t} L_{s,t} + \frac{1}{n} J \,.$$

That is, $\tilde{L}_K$ will be the regularized Laplacian of the graph with the same weights as G on K but weights $\boldsymbol{x}$ on U. In particular, $\tilde{L}_K(\boldsymbol{w}_G|_U) = \tilde{L}(\boldsymbol{w}_G)$, where $\boldsymbol{w}_G|_U$ are the weights of G restricted to the set U. We denote the graph with weights $\boldsymbol{w}_G|_K$ on K and weights $\boldsymbol{x}$ on U as $G_K(\boldsymbol{x})$

Theorem 36 *Let $G = (V, E_G, w_G)$ be a connected weighted graph with nonnegative edge weights. Let $K \sqcup U$ be a partition of $\binom{V}{2}$, i.e., all pairs of vertices. If $H = (V, E_H, w_H)$ is a connected graph such that $w_G(u, v) = w_H(u, v)$ for all $(u, v) \in K$ and $R_G(s, t) = R_H(s, t)$ for all $(s, t) \in U$, then $G = H$, i.e., $w_G(s, t) = w_H(s, t)$ for all $(s, t) \in U$.*

Proof Throughout the rest of this proof, we will restrict the domain of $\tilde{L}_K$ to the space $C_K := \{w \in \mathbb{R}_{\geq 0}^U : G_K(w) \text{ is connected}\}$. We can make this restriction as both G and H are connected by assumption. Observe that the image $\tilde{L}_K(C_K) \subset S_{++}$, as the regularized Laplacian of connected graphs are positive definite (see Theorem 5, item 3).

Now consider the composition $\log \det \tilde{L}_K : C_K \to \mathbb{R}^{n \times n}$. By Theorem 34, $\log \det$ is strictly concave within S_n^{++}. Thus, as the image of $\tilde{L}_K$ is contained in S_n^{++} and the function $\tilde{L}_K$ is affine, then the composition $\log \det \tilde{L}_K$ is strictly concave (see (Boyd and Vandenberghe, 2004, Section 3.2.2)).

Now, consider the partial derivative $(\partial \log \det \tilde{L}_K / \partial w_{s,t})(x)$ for some $(s, t) \in U$. By Theorem 35, this derivative equals $(\partial \log \det \tilde{L}_K / \partial w_{s,t})(x) = R_{G(x)}(s, t)$. In particular, for any $(s, t) \in U$, the assumption of the theorem implies

$$\frac{\partial \log \det \tilde{L}_K}{\partial w_{st}}(w_G|_U) = R_G(s, t) = R_H(s, t) = \frac{\partial \log \det \tilde{L}_K}{\partial w_{st}}(w_H|_U).$$

Thus, $\partial \log \det \tilde{L}_K / \partial w(w_G|_U) = \partial \log \det \tilde{L}_K / \partial w(w_H|_U)$. By Theorem 33, this implies that $w_G = w_H$, which means that G is equal to H. ∎

To recover the unknown part of G, we first query the effective resistance for all pairs of vertices with unknown adjacency entries. Theorem 36 ensures that these k values, together with the partially known adjacency matrix of G, uniquely determine the graph. We then search over the set of possible weights for the unknown adjacency entries to find a configuration that is consistent with our ER queries.

Theorem 37 *Let $G = (V, E, w)$ be a connected undirected weighted graph with nonnegative edge weights. Suppose that k of the edge weights of G are unknown (and the rest are known). Then there exists an algorithm to reconstruct G (i.e., find the missing k weights) using k ER queries. As a consequence, if the weights of G come from a known set of size s then the running time of the algorithm is at most $s^k \cdot \text{poly}(n)$, where $n := |V|$.*

Proof Let $K \sqcup U$ be a partition of $\binom{V}{2}$ such that $w_{s,t}$ is known for $(s, t) \in \binom{V}{2}$. Our algorithm spends $k = |U|$ ER queries to obtain the effective resistances for every pair of vertices in U.

Then, our algorithm search for edge weights of the pairs in U by exhaustively trying all s^k possibilities. For each candidate set of k weights W, our algorithm computes all pairs of effective resistances in the completion of G with W, and checks if they are consistent with the k ER values that are queried. If so, our algorithm returns W as the completion of G. By Theorem 36, there exists exactly one set of weights that are consistent with our ER queries, therefore, our algorithm will find them.

There are s^k possibilities for the edge weights in U, and computing all of the effective resistances of the graph corresponding to each of them takes $\text{poly}(n)$ time. Hence, the entire process takes $s^k \cdot \text{poly}(n)$ time. ∎

6.3. Graph completion via convex optimization

In this section, we show that the graph completion problem can be solved with k effective resistance queries by solving a convex optimization problem. As in Theorem 36, we assume there is a partition $K \sqcup U$ of $\binom{V}{2}$ such that the weights of K are known, the weights on U are unknown, and the resistances on U are known.

Recall the notation from the proof of Theorem 36 of the set $C_K := \{\boldsymbol{w} \in \mathbb{R}_{\geq 0}^U : G_K(\boldsymbol{w}) \text{ is connected}\}$. Consider the constrained optimization problem:

$$\text{minimize} \quad f(\boldsymbol{x}) = \sum_{(s,t) \in U} x_{s,t} \cdot R_G(s,t) - \log \det\left(\tilde{L}_K(\boldsymbol{x})\right) \tag{5}$$
$$\text{subject to} \quad \boldsymbol{x} \in C_K$$

We need the following lemma.

Lemma 38 *The space $C_K := \{\boldsymbol{w} \in \mathbb{R}_{\geq 0}^U : G_K(\boldsymbol{w}) \text{ is connected}\}$ is convex.*

Proof Let $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}_{\geq 0}^U$ be such that $G_K(\boldsymbol{w}_1)$ and $G_K(\boldsymbol{w}_2)$ are connected. Let $t \in [0,1]$. If $t > 0$ (respectively, $t < 1$), then all of the edges in $G_K(\boldsymbol{w}_1)$ (respectively, $G_K(\boldsymbol{w}_2)$) are contained in $G_K(t\boldsymbol{w}_1 + (1-t)\boldsymbol{w}_2)$. Thus, $G_K(t\boldsymbol{w}_1 + (1-t)\boldsymbol{w}_2)$ is connected. ∎

Theorem 39 *Let $G = (V, E_G, w_G)$ be a connected weighted graph with nonnegative edge weights. Let $K \sqcup U$ be a partition of $\binom{V}{2}$ such that the weights of K are known, the weights on U are unknown, and the resistances on U are known. The vector of unknown edge weights $w_G|_U$ is the unique solution to Problem 5.*

Proof First, we verify that this problem is convex. By Theorem 38, the feasible region C_K is convex. We therefore only need to show the objective function f is convex. The objective function f is the difference of $\sum_{(s,t) \in K} R_G(s,t) \cdot x_{st}$, which is an affine function, and the log-determinant function. For $\boldsymbol{x} \in (\mathbb{R}^{\geq 0})^U$, the matrix $\tilde{L}_K(\boldsymbol{x})$ is positive semidefinite as it is the regularized Laplacian of a graph with non-negative edge weights, so the log-determinant function is concave by Theorem 34 for all points in the domain. As f is the difference of a convex function and a concave function, it is convex.

Now we prove that a solution exists to Problem 5 and is unique. For any $\boldsymbol{x} \in C_K$, $\tilde{L}_K(\boldsymbol{x})$ is positive definite by Theorem 5. As $\log \det$ is strictly concave on the set of positive definite matrices by Theorem 34, then f is strictly convex on C_K, and Problem 5 has a unique minimizer. Finally, we know that the feasibly region C_K is non-empty as $w_G|_U \in C_K$ because we assume G is connected.

We now show that the vector of missing edge weights of G, $w_G|_U$, is the unique solution (minimizer) to Problem 5. We will use the standard fact that minimizers $\boldsymbol{x}^*$ of convex functions f are the only points that satisfy $\partial f(\mathbf{x}^*)/\partial \boldsymbol{x} = \mathbf{0}$ (see, e.g., (Boyd and Vandenberghe, 2004, Equation 4.22)).

Let $(s,t) \in U$. First, note that $\frac{\partial}{\partial x_{s,t}} \sum_{(s,t) \in U} R_G(s,t) x_{s,t} = R_G(s,t)$. Then recall that $G_K(w_G|_U)$ is exactly G. Finally, note that

$$\frac{\partial}{\partial x_{s,t}} \log \det\left(\tilde{L}_K(\boldsymbol{x})\right)\Big|_{\boldsymbol{x}} = R_{G_K(\boldsymbol{x})}(s,t)$$

by Theorem 35. Thus, $\frac{\partial}{\partial x_{s,t}} f(\mathbf{x}^*) = R_G(s,t) - R_{G_K(w_G|_U)}(s,t) = 0$, and the gradient

$$\partial f(w_G|_U)/\partial w_G|_U = \mathbf{0}.$$

Therefore, the vector of hidden edge weights $\mathbf{x}^*$ is the unique solution to Problem 5. ∎

Algorithms. The constraint that $\boldsymbol{x} \in C_K$ is equivalent to saying that $\tilde{L}(\boldsymbol{x})$ is positive definite by Theorem 5. This constraint is called a *linear matrix inequality (LMI)*. LMI constraints are common in convex programming, and problem with LMI constraints can be solved with standard algorithms for convex problems like the interior-point method (Boyd et al., 1994, Section 2.4); see also (Boyd and Vandenberghe, 2004, Section 11). Interior-point methods require a feasible point as the starting point. We can use the all-ones vector $\mathbf{1}$ as the starting point. As G is assumed to be connected and as $E_G \subseteq E_{G_K(\mathbf{1})}$, then $G_K(\mathbf{1})$ is connected and $\mathbf{1}$ is feasible.

Definition 40 *A function $f : \mathbb{R} \to \mathbb{R}$ is self concordant if, for all $x \in dom(f)$,*

$$\left| \frac{\partial^3 f(x)}{\partial x^3} \right| \leq 2 \frac{\partial^2 f(x)}{\partial x^2}^{3/2}.$$

Generally, for every $n \in \mathbb{N}$, a function $f : \mathbb{R}^n \to \mathbb{R}$ is self concordant if, for every $i \in [n]$ and for every $\mathbf{x} \in \mathbb{R}^n$, the function in a singular direction, $\tilde{f}(t) = f(\mathbf{x} + t \cdot \mathbf{1}_i)$, is self concordant.

Fact 41 ((Boyd and Vandenberghe, 2004, Example 9.5)) *The function $f(X) = -\log(\det(X))$ is self-concordant on the set of positive definite matrices S_n^{++}.*

As a corollary, we see that the function $f : \mathbb{R}^k \to \mathbb{R}$ from the equation of problem 5 is self concordant on all $\mathbf{x} \in \mathbb{R}$ such that $\tilde{L}_K(\mathbf{x}) \in S_n^{++}$, a domain which corresponds to all completions of the graph such that it is connected.

Corollary 42 *The function $f : \mathbb{R}^k \to \mathbb{R}$ defined by*

$$f(\mathbf{x}) = \sum_{(s,t) \in U} x_{s,t} \cdot R_G(s,t) - \log \det \left(\tilde{L}_K(\mathbf{x}) \right)$$

is self concordant on the domain corresponding to connected completions of G.

Proof Firstly, note that for all $\mathbf{x} \in \mathbb{R}^k$ such that $L_K(\mathbf{x})$ is the Laplacian of a *connected* graph we have that $\tilde{L}_K(\mathbf{x}) \in S_n^{++}$.

Recall that $\frac{\partial}{\partial x_{s,t}} f(\mathbf{x})$ is given by

$$\frac{\partial}{\partial x_{s,t}} f(\mathbf{x}) = R_G(s,t) - R_{G_K(\mathbf{x})}(s,t).$$

Then, the left term $R_G(s,t)$ is a constant, and will disappear when we take the second derivative $\partial^2 f / \partial x_{s,t}^2$, so this second derivative is equal to the same second derivative of $-\log \det(\tilde{L}_K(\mathbf{x}))$. Since $-\log \det(\tilde{L}_K(\mathbf{x})$ is self concordant for all $\tilde{L}_K(\mathbf{x}) \in S_n^{++}$, it follows that f is self-concordant for all completions $\mathbf{x} \in \mathbb{R}^k$ such that $L_K(\mathbf{x})$ corresponds to a connected graph. ∎

The rate of convergence for minimization problems over self-concordant functions is polynomial (see equation 9.56 of Boyd and Vandenberghe (2004)).

Acknowledgments

References

Charu C. Aggarwal. *Social Network Data Analytics*. Springer Publishing Company, Incorporated, 2014. ISBN 1489988939.

Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. *More Asymmetry Yields Faster Matrix Multiplication*, pages 2005–2039. 2025. doi: 10.1137/1.9781611978322.63. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611978322.63.

Noga Alon and Vera Asodi. Learning a hidden subgraph. *SIAM J. Discret. Math.*, 18(4): 697–712, 2005. doi: 10.1137/S0895480103431071. URL https://doi.org/10.1137/S0895480103431071.

Noga Alon, Richard Beigel, Simon Kasif, Steven Rudich, and Benny Sudakov. Learning a hidden matching. *SIAM J. Comput.*, 33(2):487–501, 2004. doi: 10.1137/S0097539702420139. URL https://doi.org/10.1137/S0097539702420139.

Dana Angluin and Jiang Chen. Learning a hidden graph using o(logn) queries per edge. *J. Comput. Syst. Sci.*, 74(4):546–556, 2008. doi: 10.1016/J.JCSS.2007.06.006. URL https://doi.org/10.1016/j.jcss.2007.06.006.

Paul Bastide and Carla Groenland. Optimal distance query reconstruction for graphs without long induced cycles. *CoRR*, abs/2306.05979, 2023. doi: 10.48550/ARXIV.2306.05979. URL https://doi.org/10.48550/arXiv.2306.05979.

Paul Bastide and Carla Groenland. Quasi-linear distance query reconstruction for graphs of bounded treelength. In Édouard Bonnet and Pawel Rzazewski, editors, *19th International Symposium on Parameterized and Exact Computation, IPEC 2024, September 4-6, 2024, Royal Holloway, University of London, Egham, United Kingdom*, volume 321 of *LIPIcs*, pages 20:1–20:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. doi: 10.4230/LIPICS.IPEC.2024.20. URL https://doi.org/10.4230/LIPIcs.IPEC.2024.20.

Zuzana Beerliova, Felix Eberhard, Thomas Erlebach, Alexander Hall, Michael Hoffmann, Matús Mihalák, and L. Shankar Ram. Network discovery and verification. *IEEE J. Sel. Areas Commun.*, 24(12):2168–2181, 2006. doi: 10.1109/JSAC.2006.884015. URL https://doi.org/10.1109/JSAC.2006.884015.

Mitchell Black, Zhengchao Wan, Gal Mishne, Amir Nayyeri, and Yusu Wang. Comparing graph transformers via positional encodings. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=va3r3hSA6n.

Hans L. Bodlaender. A note on domino treewidth. *Discret. Math. Theor. Comput. Sci.*, 3(4):141–150, 1999. doi: 10.46298/DMTCS.256. URL https://doi.org/10.46298/dmtcs.256.

Mathilde Bouvel, Vladimir Grebinski, and Gregory Kucherov. Combinatorial search on graphs motivated by bioinformatics applications: A brief survey. In Dieter Kratsch, editor, *Graph-Theoretic Concepts in Computer Science, 31st International Workshop, WG 2005, Metz, France, June 23-25, 2005, Revised Selected Papers*, volume 3787 of *Lecture Notes in Computer Science*, pages 16–27. Springer, 2005. doi: 10.1007/11604686_2. URL https://doi.org/10.1007/11604686_2.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

Arpita Ghosh, Stephen Boyd, and Amin Saberi. Minimizing effective resistance of a graph. *SIAM review*, 50(1):37–66, 2008.

Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.

Andreas Griewank, Hubertus Th. Jongen, and Man Kam Kwong. The equivalence of strict convexity and injectivity of the gradient in bounded level sets. *Mathematical Programming*, 51(1):273–278, July 1991. ISSN 1436-4646. doi: 10.1007/BF01586939. URL https://doi.org/10.1007/BF01586939.

Jotun J. Hein. An optimal algorithm to reconstruct trees from additive distance data. *Bulletin of Mathematical Biology*, 51(5):597–603, 1989. doi: 10.1007/BF02459968.

Jeremy Hoskins, Cameron Musco, Christopher Musco, and Babis Tsourakakis. Inferring networks from random walk-based node similarities. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://papers.nips.cc/paper/2018/hash/2f25f6e3.pdf.

Piotr Indyk. Sublinear time algorithms for metric space problems. In Jeffrey Scott Vitter, Lawrence L. Larmore, and Frank Thomson Leighton, editors, *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 428–434. ACM, 1999. doi: 10.1145/301250.301366. URL https://doi.org/10.1145/301250.301366.

Johan Jonasson and Oded Schramm. On the cover time of planar graphs. *Electronic Communications in Probability*, 5(none), January 2000. ISSN 1083-589X. doi: 10.1214/ecp.v5-1022. URL http://dx.doi.org/10.1214/ECP.v5-1022.

Sampath Kannan, Claire Mathieu, and Hang Zhou. Graph reconstruction and verification. *ACM Trans. Algorithms*, 14(4):40:1–40:30, 2018.

Walter Keller-Gehrig. Fast algorithms for the characteristics polynomial. *Theoretical Computer Science*, 36:309–317, 1985. ISSN 0304-3975. doi: https://doi.org/10.

1016/0304-3975(85)90049-0. URL https://www.sciencedirect.com/science/article/pii/0304397585900490.

Valerie King, Li Zhang, and Yunhong Zhou. On the complexity of distance-based evolutionary tree reconstruction. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 444–453. ACM/SIAM, 2003. URL http://dl.acm.org/citation.cfm?id=644108.644179.

Jan R. Magnus and Heinz Neudecker. *Matrix differential calculus with applications in statistics and econometrics*. Wiley series in probability and statistics. Wiley, Hoboken, NJ, third edition edition, 2019. ISBN 9781119541196.

C. St J. A. Nash-Williams. Random walk and electric currents in networks. *Mathematical Proceedings of the Cambridge Philosophical Society*, 55(02):181–194, 4 1959. ISSN 1469-8064. doi: 10.1017/S0305004100033879.

Krzysztof Onak. Testing properties of sets of points in metric spaces. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*, pages 515–526. Springer, 2008. doi: 10.1007/978-3-540-70575-8_42. URL https://doi.org/10.1007/978-3-540-70575-8_42.

Michal Parnas and Dana Ron. Testing metric properties. *Inf. Comput.*, 187(2):155–195, 2003. doi: 10.1016/S0890-5401(03)00160-3. URL https://doi.org/10.1016/S0890-5401(03)00160-3.

K. B. Petersen and M. S. Pedersen. The matrix cookbook, nov 2012. URL http://www2.compute.dtu.dk/pubdb/pubs/3274-full.html. Version 20121115.

Baron Rayleigh, John William Strutt. *The Theory of Sound*, volume 1. Macmillan and Co., London, 1877.

Lev Reyzin and Nikhil Srivastava. Learning and verifying graphs using queries with a focus on edge counting. In Marcus Hutter, Rocco A. Servedio, and Eiji Takimoto, editors, *Algorithmic Learning Theory, 18th International Conference, ALT 2007, Sendai, Japan, October 1-4, 2007, Proceedings*, volume 4754 of *Lecture Notes in Computer Science*, pages 285–297. Springer, 2007a. doi: 10.1007/978-3-540-75225-7_24. URL https://doi.org/10.1007/978-3-540-75225-7_24.

Lev Reyzin and Nikhil Srivastava. On the longest path algorithm for reconstructing trees from distance matrices. *Inf. Process. Lett.*, 101(3):98–100, 2007b. doi: 10.1016/J.IPL.2006.08.013. URL https://doi.org/10.1016/j.ipl.2006.08.013.

Guozhen Rong, Wenjun Li, Yongjie Yang, and Jianxin Wang. Reconstruction and verification of chordal graphs with a distance oracle. *Theor. Comput. Sci.*, 859:48–56, 2021. doi: 10.1016/J.TCS.2021.01.006. URL https://doi.org/10.1016/j.tcs.2021.01.006.

David E Speyer. Uniquely reconstruct a matrix m from its inverse m^{-1} if n elements of m^{-1} are unknown and n elements of m are given. MathOverflow, 2021. URL https://mathoverflow.net/q/407563.

Daniel Spielman. Spectral and algebraic graph theory. Available at https://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf, 2025.

Daniel A. Spielman. Trees and distances, 2012. University Lecture.

William Thomson and Peter Tait. *Treatise on Natural Philosophy*. Oxford University Press, United Kingdom, 1867.

Gunther Uhlmann. Electrical impedance tomography and calderón's problem. *Inverse problems*, 25 (12):123011, 2009.

Gunther Uhlmann. 30 years of calderón's problem. In *Séminaire Laurent Schwartz — EDP et applications*, number 13, page 25. 2012. doi: 10.5802/slsedp.40. URL http://www.numdam.org/articles/10.5802/slsedp.40/.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Ameya Velingker, Ali Sinop, Ira Ktena, Petar Veličković, and Sreenivas Gollapudi. Affinity-aware graph networks. *Advances in Neural Information Processing Systems*, 36, 2024.

Wikipedia contributors. Electrical resistivity tomography — Wikipedia, The Free Encyclopedia, 2024. URL https://en.wikipedia.org/wiki/Electrical_resistivity_tomography. Accessed: 2024-01-24.

Dominik M. Wittmann, Daniel Schmidl, Florian Blöchl, and Fabian J. Theis. Reconstruction of graphs based on random walks. *Theor. Comput. Sci.*, 410(38-40):3826–3838, 2009. doi: 10.1016/J.TCS.2009.05.026. URL https://doi.org/10.1016/j.tcs.2009.05.026.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Bohang Zhang, Shengjie Luo, Liwei Wang, and Di He. Rethinking the expressive power of GNNs via graph biconnectivity. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=r9hNv76KoT3.