Fast Estimation and Optimization of Resistance Diameter on Graphs

Anonymous Author(s)

ABSTRACT

The resistance diameter of a graph is the maximum resistance distance among all pairs of nodes in the graph, which has found various applications in many scenarios. However, direct computation of resistance diameter involves the pseudoinverse of graph Laplacian, which takes cubic time and is thus infeasible for huge networks with millions of nodes. In this paper, we consider the computation and optimization problems for resistance diameter of a graph. First, we develop a nearly linear time algorithm to approximate the resistance diameter, which has a theoretically guaranteed error. Then, we propose and study an optimization problem of adding a fixed number of edges to a graph, such that the resistance diameter of the resulting graph is minimized. We show that this optimization problem is NP-hard, and that the objective function is non-supermodular but monotone. Moreover, we propose two fast heuristic algorithms to approximately solve this problem. Finally, we conduct extensive experiments on different networks with sizes up to one million nodes, demonstrating the superiority of our algorithms in terms of efficiency and effectiveness.

KEYWORDS

Resistance distance, Resistance diameter, Combinatorial optimization, Graph mining, Linear algorithm, Convex hull, Laplacian solver

1 INTRODUCTION

Effective resistance, as a fundamental graph metric, has found wide-ranging theoretical and practical applications. In theory, resistance distance has become a cornerstone of algorithmic graph theory, leading to breakthroughs in fast algorithms for problems such as spectral graph sparsification [57], maximum flow approximation [16], random spanning trees generation [41], maximizing the number of spanning trees [36], and solving traveling salesman problems [4]. In practical scenarios, resistance distance has been applied to diverse fields like collaborative recommendation [21], graph embedding [11], image segmentation [7], mining important nodes and edges [10, 37, 55], and link prediction [42]. In data management, it has been used in graph processing systems [52], density-based clustering [56], and localizing anomalous changes [58]. Recently, new methods for evaluating resistance distance have emerged [39, 65], underscoring its significance in both theory and application [19].

Alongside resistance distance, shortest path distance is another key graph metric, defined as the length of the shortest path between two nodes. The diameter of a graph, which is the maximum shortest-path distance, is an important measure with applications in graph analytics [70], privacy preservation [47], CPU cache efficiency [62], community search [40], and community discovery [17]. While shortest path distance focuses on a single path, resistance distance accounts for all paths between two nodes, often leading to more effective applications [10, 23]. For instance, current flow closeness centrality based on resistance distance shows greater discrimination power than its shortest-path counterpart [37]. A key measure based on resistance distance is resistance diameter, which has numerous applications, including clustering in graphs [3] and hypergraphs [2], distributed clock synchronization over wireless networks [28], and minimizing the Kirchhoff index [30]. However, computing resistance diameter for large graphs is computationally expensive, and efficient algorithms for this task are still lacking. Moreover, reducing resistance diameter, for example by adding edges, is of significant interest in many applications where a smaller resistance diameter is desirable.

In this paper, we study the computation and optimization problems for the resistance diameter $R(\mathcal{G})$. The main contributions and work of this paper are summarized as follows.

- We propose a fast ϵ-approximation algorithm FastRD to evaluate the resistance diameter $R(\mathcal{G})$ of a graph $\mathcal{G}$, which is based on the Johnson-Lindenstrauss Lemma, Laplacian solvers and convex hull. The algorithm has a nearly linear time complexity $\widetilde{O}\left((m+nl)/\epsilon^2\right)$, where the $\tilde{O}(\cdot)$ notation suppresses the poly$(\log n)$ factors, $\epsilon > 0$ is the error parameter, and l is the number of nodes in the boundary of the approximated convex hull.
- We propose and study the following optimization problem: Given a graph $\mathcal{G}$ and an integer $k \ll n$, how to select a set P of k edges in set $Q = (V \times V)\backslash E$ to add to $\mathcal{G}$ such that the resistance diameter of the resulting graph is minimized. We show that this problem is NP-hard, and that the objective function is not supermodular.
- We propose two efficient greedy heuristic algorithms MinDiaEi and MinDiaCH to approximately solve the combinatorial optimization problem. The former has time complexity $\widetilde{O}(km)$, while the latter has complexity $\widetilde{O}\left(k(m+nl)/\epsilon^2\right)$ for a given error parameter $\epsilon > 0$.
- We execute extensive numerical experiments on a large variety of real and model networks to evaluate the performance of our proposed algorithms: FastRD for computing resistance diameter, and MinDiaEi and MinDiaCH for minimizing resistance diameter by edge addition. The obtained experimental results demonstrate that our three approximation algorithms are efficient and accurate. Particularly, they are scalable to large graphs with over one million nodes. Moreover, MinDiaEi and MinDiaCH outperform several baseline strategies for edge selection.

2 RELATED WORK

In this section, we briefly review the literature related to our work. The resistance diameter can be seen as an extension of the shortest-path-based diameter, as they are equivalent in tree-like graphs. The diameter of a graph represents the greatest distance between any two nodes, reflecting the maximum time needed for information or matter to propagate. Graph diameter has found applications in data management and mining, including route recommendation on road networks [66] and reachability querying in large graphs [15]. Significant efforts have been made to design efficient algorithms for computing graph diameter [38, 61], as well as strategies for optimizing it, such as reducing graph diameter by adding edges [1, 18, 22]. However, since resistance diameter accounts for all paths between nodes, methods for computing or optimizing shortest-path diameter do not directly apply to resistance diameter.

Fast computation of resistance distance is essential for many applications, prompting the development of various algorithms, such as random projection methods [43, 57], accelerated by Wilson's algorithm [29]. Local algorithms using random walks and spanning trees [48], as well as Monte Carlo approaches [39, 65], have been proposed to improve performance. While these methods are efficient for computing node-to-node resistance distances, computing resistance diameter, which involves calculating distances for $O(n^2)$ node pairs, remains impractical for large graphs. Although there is no closed-form expression for the resistance diameter of general graphs, explicit formulas have been derived for special structures like hierarchical graphs [50], Hanoi graphs [54], Sierpiński gaskets [31], and Koch networks [69, 73]. Additionally, the scaling behavior of resistance diameter has been studied across various graph topologies, showing linear growth in path and cycle graphs [67], sub-linear growth in hierarchical graphs [54], and logarithmic growth in complete trees and torus graphs [8, 75]. Interestingly, for most real-world networks, resistance diameter and average resistance distance do not depend on the number of nodes [64, 68].

In terms of optimization, prior work has focused on minimizing resistance distance or the Kirchhoff index by edge addition [27]. Researchers have developed algorithms to minimize resistance distance between specific node pairs [13] and to optimize the Kirchhoff index [51, 59]. However, no previous methods directly address minimizing resistance diameter by adding edges, despite edge addition being widely used in other graph editing applications, such as improving node centrality [55] or maximizing overall opinion dynamics [74]. Thus, minimizing resistance diameter through graph edits remains an open and challenging problem.

3 PRELIMINARIES

3.1 Graph and Related Matrices

Let $\mathcal{G} = (V, E)$ denote a connected undirected unweighted graph with node/vertex set V and edge set E. The numbers of nodes and edges in $\mathcal{G}$ are $n = |V|$ and $m = |E|$, respectively. For graph $\mathcal{G} = (V, E)$, let $Q = (V \times V)\backslash E$ be the set of edges that are nonexistent in $\mathcal{G}$. For a nonempty edge set $A \subset Q$, we use $\mathcal{G}(A) = (V, E \cup A)$ to denote the augmented graph of $\mathcal{G} = (V, E)$ with the same node set V as $\mathcal{G}$ but more edges than $\mathcal{G}$.

All the eigenvalues of Laplacian matrix $\boldsymbol{L}$ are non-negative, with zero being the unique eigenvalue. Let $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$ denote its n eigenvalues, and let $\boldsymbol{u}_k$, $k = 1, 2, \ldots, n$, stand for their corresponding mutually orthogonal unit eigenvectors. The second smallest eigenvalue λ_2 of matrix $\boldsymbol{L}$ is often called the algebraic connectivity of the graph $\mathcal{G}$ and denoted as $\lambda_2(\mathcal{G})$, and its corresponding normalized eigenvector is called the Fiedler vector of $\mathcal{G}$. Laplacian matrix $\boldsymbol{L}$ can be decomposed as $\boldsymbol{L} = \sum_{k=1}^{n} \lambda_k \boldsymbol{u}_k \boldsymbol{u}_k^\top$. Note that $\boldsymbol{L}$ is singular, since it has a eigenvalue 0. Thus, $\boldsymbol{L}$ cannot be inverted. Alternatively, we use the Moore-Penrose generalized inverse of $\boldsymbol{L}$, which is simply called pseudo-inverse of $\boldsymbol{L}$ [9]. Let $\boldsymbol{L}^\dagger$ denote the pseudo-inverse of $\boldsymbol{L}$, which can be expressed as $\boldsymbol{L}^\dagger = \sum_{k=2}^{n} \frac{1}{\lambda_k} \boldsymbol{u}_k \boldsymbol{u}_k^\top$.

The symmetry of both matrices $\boldsymbol{L}$ and its pseudo-inverse $\boldsymbol{L}^\dagger$ implies that $\boldsymbol{L}$ and $\boldsymbol{L}^\dagger$ share identical null space [9], obeying $\boldsymbol{L}\boldsymbol{1} = \boldsymbol{0}$ and $\boldsymbol{L}^\dagger\boldsymbol{1} = \boldsymbol{0}$. Consider the relation $\mathcal{J} = \boldsymbol{1}\boldsymbol{1}^\top$, we obtain $\boldsymbol{L}\mathcal{J} = \mathcal{J}\boldsymbol{L} = \boldsymbol{L}^\dagger\mathcal{J} = \mathcal{J}\boldsymbol{L}^\dagger = \boldsymbol{O}$. Applying the the spectral decompositions for matrices $\boldsymbol{L}$ and $\boldsymbol{L}^\dagger$, the pseudoinverse $\boldsymbol{L}^\dagger$ of $\boldsymbol{L}$ can be obtained to be [27]

$$\boldsymbol{L}^\dagger = \left(\boldsymbol{L} + \frac{1}{n}\mathcal{J}\right)^{-1} - \frac{1}{n}\mathcal{J}. \tag{1}$$

3.2 Electrical Networks, Resistance Distance and Resistance Diameter

By replacing every edge in graph $\mathcal{G}$ with a unit resistance, we obtain an electrical network [20] associated with graph $\mathcal{G}$. In the case incurring no confusion, for a graph $\mathcal{G}$, we also use $\mathcal{G}$ to denote its corresponding electrical network. The resistance distance $r(u, v)$ between two nodes u and v in graph $\mathcal{G} = (V, E)$ is defined as the effective resistance between u and v in the corresponding electrical network [33], which can be expressed in terms of the entries of matrix $\boldsymbol{L}^\dagger$ as

$$r(u, v) = \boldsymbol{L}^\dagger_{uu} + \boldsymbol{L}^\dagger_{vv} - 2\boldsymbol{L}^\dagger_{uv}. \tag{2}$$

For a graph $\mathcal{G} = (V, E)$, the maximum value of resistance distances over all its node pairs is called the resistance diameter and is denoted by $R(\mathcal{G})$. That is,

$$R(\mathcal{G}) = \max_{u,v \in V} r(u, v). \tag{3}$$

4 ALGORITHMS FOR COMPUTING RESISTANCE DIAMETER

In this section, we introduce three algorithms for computing resistance diameter.

4.1 Exact Computation by Matrix Inverse

By using (1), (2), and (3), we can compute the exact resistance diameter for any graph $\mathcal{G}$. First, using (1) we obtain the pseudoinverse $\boldsymbol{L}^\dagger$ of matrix $\boldsymbol{L}$ by inverting the matrix $\boldsymbol{L} + \frac{1}{n}\mathcal{J}$ in $O(n^3)$ time. Then, according to (2), we compute the resistance distance $r(u, v)$ for all the $O(n^2)$ pairs of nodes u and v in $\mathcal{G}$, which takes time $O(n^2)$. Finally, we use (3) to find the resistance diameter $R(\mathcal{G})$ of graph $\mathcal{G}$, which runs in $O(n^2)$ time. The rigorous algorithm is called Exact, which is described in Algorithm 1. It is obvious that the total running time of Exact is $O(n^3)$.

4.2 Algorithm Based on JL Lemma and Laplacian Solver

As shown above, the most time-consuming step in the Algorithm 1 for computing resistance diameter is inverting the matrix $\boldsymbol{L} + \frac{1}{n}\mathcal{J}$, which requires $O(n^3)$ time. To avoid this time-consuming matrix inverse operation, here we describe a randomized algorithm, which approximates the resistance diameter in $O(n^2)$ time.

Algorithm 1: Exact($\mathcal{G}$)

```
Input : A connected graph $\mathcal{G} = (V, E)$ with Laplacian matrix $\boldsymbol{L}$
Output : $R(\mathcal{G})$ : The resistance diameter of graph $\mathcal{G}$
1 Compute the pseudoinverse $\boldsymbol{L}^\dagger$ of $\boldsymbol{L}$ by
    $\boldsymbol{L}^\dagger = \left(\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}\right)^{-1} - \frac{1}{n}\boldsymbol{J}$
2 Compute $r(u, v)$ for all node pairs in $\mathcal{G}$ by
    $r(u, v) = \boldsymbol{L}^\dagger_{uu} + \boldsymbol{L}^\dagger_{vv} - 2\boldsymbol{L}^\dagger_{uv}$
3 $R(\mathcal{G}) = \max_{u,v \in V} r(u, v)$
4 return $R(\mathcal{G})$
```

Before introducing our algorithm, we first recast the resistance distance $r(u, v)$ as [57]

$$r(u, v) = \left\|\boldsymbol{B}\boldsymbol{L}^\dagger\left(\boldsymbol{e}_u - \boldsymbol{e}_v\right)\right\|_2^2. \qquad (4)$$

In other words, we can embed the n nodes in graph $\mathcal{G}$ to n points corresponding to n m-dimension vectors $\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$, $i = 1, 2, \ldots, n$, in Euclidean space $\mathbb{R}^m$, which preserve the resistance distances. For two vectors $\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_u$ and $\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_v$ in $\mathbb{R}^m$, let $d(u, v)$ denote their Euclidean distance. Then, we have $r(u, v) = d(u, v)^2$.

Equation (4) reduces the computation of resistance distance $r(u, v)$ to the calculation of the ℓ_2 norms $\left\|\boldsymbol{B}\boldsymbol{L}^\dagger\left(\boldsymbol{e}_u - \boldsymbol{e}_v\right)\right\|_2^2$ of two vectors in $\mathbb{R}^m$. However, the complexity for exactly computing this ℓ_2 norms is still high, since the dimension m of vectors $\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$ ($i = 1, 2, \ldots, n$) is high and it still needs inverting matrix $\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}$ to obtain $\boldsymbol{L}^\dagger$.

We first alleviate the computation burden by reducing the dimension of vectors. For this purpose, we apply the Johnson-Lindenstrauss (JL) Lemma [32] to approximate the ℓ_2 norms. For $\left\|\boldsymbol{B}\boldsymbol{L}^\dagger\left(\boldsymbol{e}_u - \boldsymbol{e}_v\right)\right\|_2^2$, if we project the set of n m-dimension vectors, i.e., the n column vectors of matrix $\boldsymbol{B}\boldsymbol{L}^\dagger$, onto a low d-dimension subspace spanned by the columns of a random matrix $\boldsymbol{Q} \in \mathbb{R}^{d\times m}$ with entries being $\pm 1/\sqrt{d}$, where $d = \left\lceil 24\log(n)/\epsilon^2\right\rceil$ for given ϵ, then we get an ϵ-approximation of $\left\|\boldsymbol{B}\boldsymbol{L}^\dagger\left(\boldsymbol{e}_u - \boldsymbol{e}_v\right)\right\|_2^2$ with high probability. For consistency, we introduce the JL Lemma [32].

Lemma 4.1. *(JL Lemma) Given fixed vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n \in \mathbb{R}^m$ and $\epsilon > 0$, let $\boldsymbol{Q}_{d\times m}$, $d \geq 24\log n/\epsilon^2$, be a matrix, each entry of which is equal to $1/\sqrt{d}$ or $-1/\sqrt{d}$ with the same probability $1/2$. Then with probability at least $1 - 1/n$,*

$$(1-\epsilon)\left\|\boldsymbol{v}_i - \boldsymbol{v}_j\right\|_2^2 \leq \left\|\boldsymbol{Q}\boldsymbol{v}_i - \boldsymbol{Q}\boldsymbol{v}_j\right\|_2^2 \leq (1+\epsilon)\left\|\boldsymbol{v}_i - \boldsymbol{v}_j\right\|_2^2$$

for all pairs $i, j \leq n$.

Let $\boldsymbol{Q}_{d\times m}$ be a random $\pm 1/\sqrt{d}$ matrix where $d = \left\lceil 24\log(n)/\epsilon^2\right\rceil$. By Lemma 4.1, for any $u, v \in V$ we have

$$r(u, v) \overset{\epsilon}{\approx} \left\|\boldsymbol{Q}\boldsymbol{B}\boldsymbol{L}^\dagger\left(\boldsymbol{e}_u - \boldsymbol{e}_v\right)\right\|_2^2. \qquad (5)$$

Thus, if we embed the n vertices in graph $\mathcal{G}$ to n vectors $\boldsymbol{Q}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$, $i = 1, 2, \ldots, n$, in a low-dimension space $\mathbb{R}^d$, the resistance distance for any node pair is approximately preserved.

However, the computation of the n vectors $\boldsymbol{Q}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$ ($i = 1, 2, \ldots, n$) still requires the pseudoinverse $\boldsymbol{L}^\dagger$, which involves inverting matrix $\boldsymbol{L} + \frac{1}{n}\boldsymbol{J}$. In order to avoid matrix inversion, one can resort to the fast symmetric, diagonally-dominant (SDD) linear system solver [35] to evaluate $\boldsymbol{Q}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$, since $\boldsymbol{L}$ is an SDDM matrix. In the sequel, for the convenience of description, we use the notation $\tilde{O}(\cdot)$ to hide poly log factors.

Algorithm 2: ApproxRD($\mathcal{G}, \epsilon$)

```
Input : A connected graph $\mathcal{G} = (V, E)$, a parameter $\epsilon$
Output : $\bar{R}(\mathcal{G})$ : The approximate resistance diameter of graph $\mathcal{G}$
1 $d = \lceil 24\log n/\epsilon^2\rceil$
2 $\tilde{\boldsymbol{X}}_{d\times n} \leftarrow$ ApproxER$(\mathcal{G}, \epsilon)$
3 Compute $\tilde{r}(u, v)$ for all node pairs in $\mathcal{G}$ by
    $\tilde{r}(u, v) = ||\tilde{\boldsymbol{X}}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$
4 $\bar{R}(\mathcal{G}) \leftarrow \max_{u,v\in V} ||\tilde{\boldsymbol{X}}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$
5 return $\bar{R}(\mathcal{G})$
```

Lemma 4.2. *There is an algorithm $\boldsymbol{x} = LaplSolve\left(\boldsymbol{L}, \boldsymbol{z}, \delta\right)$ which takes a Laplacian matrix $\boldsymbol{L}$, a column vector $\boldsymbol{z}$, and an error parameter $\delta > 0$, and returns a column vector $\boldsymbol{x}$ satisfying $\boldsymbol{1}^\top\boldsymbol{x} = 0$ and*

$$\left\|\boldsymbol{x} - \boldsymbol{L}^\dagger\boldsymbol{z}\right\|_{\boldsymbol{L}} \leq \delta\left\|\boldsymbol{L}^\dagger\boldsymbol{z}\right\|_{\boldsymbol{L}}.$$

The algorithm runs in expected time $\widetilde{O}(m\log(1/\delta))$.

Based on the Laplacian solvers and the JL Lemma, an approximation algorithm ApproxER was proposed in [57] to estimated resistance distance in nearly linear time with respect to the number of edges, as stated in Lemma 4.3.

Lemma 4.3. *There is an $\widetilde{O}\left(m\log(1/\delta)/\epsilon^2\right)$-time algorithm which on input $\epsilon > 0$, $\delta \leq \frac{\epsilon}{3}\sqrt{\frac{2(1-\epsilon)}{(1+\epsilon)n^3}}$ and $\mathcal{G} = (V, E)$ computes a $\lceil 24\log n/\epsilon^2\rceil \times n$ matrix $\tilde{\boldsymbol{X}}$ such that with probability at least $1 - 1/n$,*

$$r(u, v) \overset{\epsilon}{\approx} ||\tilde{\boldsymbol{X}}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$$

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

By using Lemma 4.3, we develop an approximation algorithm ApproxRD estimating the resistance diameter $R(\mathcal{G})$ for graph $\mathcal{G}$ in $O(n^2)$ time. The pseudocode of ApproxRD is provided in Algorithm 2.

4.3 Fast Algorithm Based on Convex Hull

The above approximation algorithm ApproxER based on the Laplacian solvers and the JL Lemma is still infeasible for large networks, since it runs in $O(n^2)$ time. This is caused by the computation of the resistance distances for all the $O(n^2)$ pairs of nodes in $\mathcal{G}$. In this subsection, we will show that one only need to compute the resistance distances for a small number of node pairs, since we are only concerned with the resistance diameter $R(\mathcal{G})$. We then develop a fast algorithm, which approximates $R(\mathcal{G})$ of an arbitrary graph $\mathcal{G}$. Our algorithm has a nearly linear time and space cost with respect to the number of edges. Furthermore, it has a guaranteed theoretical error with high probability.

Equation (5) reduces the approximation estimation of resistance distances to the calculation of the distances between the $O(n^2)$ pairs of n points $\boldsymbol{Q}\boldsymbol{B}\boldsymbol{L}^\dagger\boldsymbol{e}_i$ ($i = 1, 2, \ldots, n$) in Euclidean space $\mathbb{R}^d$. To calculate the resistance diameter $R(\mathcal{G})$, we need to find the pair of nodes with the largest resistance distance. Equation (5) shows that we only to find the distance of the pair of farthest points in d-dimensional Euclidean space, which can be solved by the convex hull technique.

Definition 4.4. *[49] Given a set of n points $S = \{v_1, v_2, \dots, v_n\}$ in $\mathbb{R}^d$, its convex hull is the (unique) minimal convex polytope containing S.*

For a given convex hull of point set S, its boundary is denoted by $C(S)$, and the subset of points in S, which lie on the boundary of the convex hull is denoted by $\bar{S}$. By Definition 4.4, it can be observed that in an Euclidean space the pair of nodes with the maximum distance in a point set must lie on the boundary of the convex hull corresponding to that point set.

There are many methods to find the convex hull of a point set of n points in a low-dimensional Euclidean space. However, the determination of the samples in the convex hull of a point set of high dimensions is a time-complex task. For dimensions $d > 3$, the time for computing the convex hull is $O(n^{\lfloor d/2 \rfloor})$, matching the worst-case output complexity of the problem [14]. Therefore, computing the convex hull of n points in d-dimension space is time-consuming. Fortunately, there is a fast algorithm ApproxCH approximating the convex hull [5]. Before introducing the algorithm ApproxCH, we provide some more definitions. For any given set $S = \{v_i \in \mathbb{R}^d : i = 1, 2, \dots, n\}$, let $D(S)$ denote the diameter of S. That is, $D(S) = \max_{v_i, v_j \in S} ||v_i - v_j||_2$, which is the maximum distance between all pairs of points v_i and v_j, obeying relation $D(S) = D(\bar{S})$.

Lemma 4.5. *[5] There is an algorithm $\hat{S} = \text{ApproxCH}(S, \theta)$ which takes a set $S = \{v_i \in \mathbb{R}^d : i = 1, 2, \dots, n\}$ and an error parameter $\theta \in (0, 1)$, and returns an l-node subset $\hat{S}$ of $\bar{S}$. The algorithm runs in $O(nl(d + \theta^{-2}))$ time, and the Euclidean distance for any $p \in \bar{S}$ to $C(\hat{S})$ is at most $\theta D(S)$, where $C(\hat{S})$ is the boundary of convex hull of $\hat{S}$.*

Lemma 4.5 shows that using the ApproxCH algorithm, we can obtain an approximate point set $\hat{S}$ for $\bar{S}$, the boundary of convex hull $C(S)$ in a d-dimensional Euclidean space. Moreover, ApproxCH provides an upper bound for the distance between any point in $\bar{S}$ and any point in $C(\hat{S})$. Based on Lemma 4.5, we can show that $D(\hat{S})$ is a good approximation for $D(\bar{S})$, as stated in Lemma 4.6.

Lemma 4.6. *Given a point set $S = \{v_i \in \mathbb{R}^d : i = 1, 2, \dots, n\}$, a parameter $\theta = \frac{\epsilon}{12}$, a subset $\bar{S} \subseteq S$, whose points lie on the boundary of the convex hull $C(S)$ of set S, and $\hat{S} = \text{ApproxCH}(S, \theta)$ that is a l-node subset of $\bar{S}$. Let $x, y \in \hat{S}$, $u, v \in \bar{S}$, $d(x, y) = D(\hat{S})$, and $d(u, v) = D(\bar{S})$. Then, we have*

$$d(x, y) \overset{\frac{\epsilon}{6}}{\approx} d(u, v). \tag{6}$$

Making use of Eq. (4), Lemma 4.3, and Lemma 4.6, we obtain the following result.

Lemma 4.7. *Given a graph $\mathcal{G} = (V, E)$, a set $S = \{v_i \in \mathbb{R}^m : i = 1, 2, \dots, n\}$, where $v_i = \boldsymbol{QBL}^{\dagger}\boldsymbol{e}_i$, a parameter $\epsilon > 0$, and $\theta = \frac{\epsilon}{12}$. Suppose that $\hat{S} = \text{ApproxCH}(S, \theta)$ is an l-point subset of $\bar{S}$ and $\bar{R}(\mathcal{G}) = \text{ApproxRD}(\mathcal{G}, \epsilon)$. Let $\hat{R}(\mathcal{G}) = D(\hat{S})^2 = \max_{u,v \in \hat{S}} d(u, v)^2$. Then, we have*

$$\hat{R}(\mathcal{G}) \overset{\epsilon/3}{\approx} \bar{R}(\mathcal{G}). \tag{7}$$

Algorithm 3: FastRD($\mathcal{G}, \epsilon$)

```
Input  : A connected graph $\mathcal{G} = (V, E)$, a parameter $\epsilon$
Output : $\hat{R}(\mathcal{G})$: Approximation of resistance diameter
         $R(\mathcal{G})$ of graph $\mathcal{G}$
1 $d = \lceil 24 \log n / \epsilon^2 \rceil$, $\theta = \frac{\epsilon}{12}$
2 $\tilde{X}_{d \times n} \leftarrow$ ApproxER($\mathcal{G}, \epsilon$)
3 $S \leftarrow \{s_i \in \mathbb{R}^d | s_i = \tilde{X}_{[:,i]}, i = 1, 2, \dots, n\}$
4 $\hat{S} \leftarrow$ ApproxCH($S, \theta$)
5 Compute $\tilde{r}(u, v)$ for all node pairs in $\hat{S}$ by
      $\tilde{r}(u, v) = ||\tilde{X}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$
6 $\hat{R}(\mathcal{G}) \leftarrow \max_{u,v \in \hat{S}} \tilde{r}(u, v)$
7 return $\hat{R}(\mathcal{G})$
```

Since $\hat{S}$ contains l points, we can obtain $\hat{R}(\mathcal{G})$ by computing the distances between l^2 pairs of points, instead of $O(n^2)$ pairs of points. In general, l is much smaller than n, using Lemma 4.7 to evaluate the resistance diameter of a graph significantly reduces the computation time.

Based on the above results, we are now in position to propose a fast algorithm FastRD approximating the resistance diameter of $\mathcal{G}$, the pseudocode of which is presented in Algorithm 3. In FastRD, $\tilde{X}$ is a $\lceil 24 \log n / \epsilon^2 \rceil \times n$ matrix, S is a set of points, which are the column vectors of matrix $\tilde{X}$. Set $\hat{S}$ is an approximation of $\bar{S}$, which is the set of points on the boundary of the convex hull $C(S)$. The performance of the algorithm FastRD is given in Theorem 4.8.

Theorem 4.8. *The algorithm FastRD($\mathcal{G}, \epsilon$) runs in $\widetilde{O}\left((m + nl)/\epsilon^2\right)$ time, and outputs an approximation value $\hat{R}(\mathcal{G})$ of resistance diameter $R(\mathcal{G})$ for graph $\mathcal{G}$, satisfying*

$$(1 - \epsilon)R(\mathcal{G}) \leq \hat{R}(\mathcal{G}) \leq (1 + \epsilon)R(\mathcal{G}).$$

5 MINIMIZING RESISTANCE DIAMETER BY EDGE ADDITION

In this section, we formulate and study the problem of adding a fixed number edges in order to minimize the resistance diameter of a graph.

5.1 Problem Statement, Optimal Solution, and Simple Greedy Algorithm

It is well-known that [13], adding an edge to a graph does not increase the resistance distance between any pair of nodes, including the resistance diameter. This motivates us to study the problem of how to choose a fixed number k of nonexistent edges to a graph in order to minimize the resistance diameter of the resultant new graph. The problem is mathematically formulated as follows.

Problem 1. *(Resistance Diameter Minimization) Given a graph $\mathcal{G} = (V, E)$ with n nodes and m edges and a candidate edge set $Q = (V \times V) \backslash E$, for any integer $1 \leq k \leq |Q|$, find an edge subset*

```
Algorithm 4: SIMPLE($\mathcal{G}, Q, k$)
  Input  : A connected graph $\mathcal{G} = (V, E)$, a candidate edge
           set $Q$, an integer $1 \le k \le |Q|$
  Output : $P$ : A subset of $Q$ with $|P| = k$ edges
1 Initialize solution $P = \emptyset$
2 for $i = 1$ to $k$ do
3     Select $e_i$ s.t. $e_i \leftarrow \arg\min_{e \in Q \setminus P}$ EXACT $(\mathcal{G}(\{e\}))$
4     Update solution $P \leftarrow P \cup \{e_i\}$
5     Update the graph $\mathcal{G} \leftarrow \mathcal{G}(\{e_i\})$
6 return $P$
```

$P^* \subseteq Q$ such that the resistance diameter of $\mathcal{G}(P^*) = (V, E \cup P^*)$ is minimized, that is,

$$P^* \in \arg\min_{P \subseteq Q, |P|=k} R(\mathcal{G}(P)).$$

Problem 1 is a intrinsically combinatorial one subject to a cardinality constraint. We can obtain its optimal solutions by exhausting all the $\binom{|Q|}{k}$ subsets P with k edges. For each subset, we calculate the resistance diameter of the resultant graph, which requires $O(n^3)$ time. Then, output the optimal solution, which minimizes the resistance diameter. Although this method is simple, it is computationally impossible even for small networks, since its has an exponential complexity $O(\binom{|Q|}{k} n^3)$.

To tackle the exponential complexity of brute-force search, one often resorts to greedy heuristics. Below we present a simple greedy algorithm for Problem 1, which is outlined in Algorithm 4 and described as follows. Initially, we set the set P of added edges to be empty, then k edges are added from set $Q \setminus P$ iteratively. In each iteration step i, edge e_i in set $Q \setminus P$ of candidate edges is selected, which minimizes the resistance diameter of the new graph. The algorithm terminates when k edges are selected to be added to P. For every candidate edge, it needs computing resistance diameter of a resultant graph. A direct calculation of resistance diameter requires $O(n^3)$ time, leading to a total computation complexity $O(k|Q|n^3)$.

We also address the hardness of the resistance diameter minimization problem and the non-supermodular property of the objective function, with additional details provided in the appendix.

5.2 Two Fast Approximation Algorithms

Although the simple greedy algorithm in Algorithm 4 is much faster than the brute-force algorithm, it is not applicable to large-scale networks since it takes too much time to calculate $R(\mathcal{G}(\{e\}))$ for each edge $e \in Q \setminus P$ in cubic time $O(n^3)$ in each iteration. Then, it take quadratic time $O(n^2)$ to obtain the edge $Q \setminus P$ that minimizes the resistance diameter of resultant graph. To reduce the computational complexity, we propose two nearly linear time approximation algorithms. The former iteratively find k edges to minimize an upper bound of the resistance diameter, while the latter finds the furthest k pairs of nodes and add edges among them.

5.2.1 Fast Alternative Algorithm Minimizing an Upper Bound of Resistance Diameter. We first approximately solves Problem 1 from the perspective of spectral graph theory. To this end, we propose a fast approximation algorithm to minimize an upper bound of the resistance diameter, as an alternative way to address Problem 1. Note that the approach of optimizing bounds of a quantity, rather than the quantity itself, has been previously explored in the literature [44, 60]. Since $2/\lambda_2$ is an upper bound for the resistance diameter $R(\mathcal{G})$[63], Problem 1 can be reduced to maximizing λ_2 by adding k nonexistent edges to graph $\mathcal{G}$.

```
Algorithm 5: MINDIAEI($\mathcal{G}, Q, k, \delta$)
  Input  : A connected graph $\mathcal{G} = (V, E)$, a candidate edge
           set $Q$, an integer $1 \le k \le |Q|$, a parameter $\delta$
  Output : $P$ : A subset of $Q$ with $|P| = k$ edges
1 Initialize solution $P = \emptyset$
2 for $i = 1$ to $k$ do
3     Let $\boldsymbol{L}$ be the Laplacian matrix of graph $\mathcal{G}$
4     Compute $(\lambda, \boldsymbol{u})$ = EIGENPAIR$(\boldsymbol{L}, \delta)$
5     Select $e_i$ s.t. $e_i \leftarrow \arg\max_{(j,t) \in Q \setminus P} (\boldsymbol{u}_j - \boldsymbol{u}_t)^2$
6     Update solution $P \leftarrow P \cup \{e_i\}$
7     Update the graph $\mathcal{G} \leftarrow \mathcal{G}(\{e_i\})$
8 return $P$
```

Note that the problem of maximizing λ_2 by adding k edges is combinatorial and can be solved exactly by exhaustive search. It is easy to see that there are $\binom{|Q|}{k}$ choices for edge selection. For each of $\binom{|Q|}{k}$ cases, we compute λ_2 of the corresponding Laplacian matrix and return the set of k edges maximizing λ_2. However, this is not practical for large $|Q|$ and k. Therefore, we use the efficient heuristic algorithm proposed in [26]. Concretely, this heuristic method adds the k edges one at a time. At each time, it chooses the edge (i, j) which has the largest value of $(\boldsymbol{u}_i - \boldsymbol{u}_j)^2$, where $\boldsymbol{u}$ is a Fiedler vector of the current graph. Since directly computing the Fiedler vector requires $O(n^3)$ time, we need to explore an efficient method for computing Fiedler vector quickly. Fortunately, this can be solved by the method in [6], as given in the following lemma.

LEMMA 5.1. *[6] For a connected graph $\mathcal{G}$ with the Laplacian matrix $\boldsymbol{L}$, there exists a nearly-linear time $\widetilde{O}(m)$ algorithm $(\lambda, \boldsymbol{u})$ = EIGENPAIR$(\boldsymbol{L}, \delta)$, whose outputs λ and $\boldsymbol{u}$ are, respectively, approximations for algebra connectivity and Fiedler vector for graph $\mathcal{G}$, satisfying $\lambda = \boldsymbol{u}^\top \boldsymbol{L} \boldsymbol{u} \le (1+\delta)\lambda_2$.*

Based on Lemma 5.1, we propose a fast algorithm called MINDIAEI to approximately solve Problem 1. The outline of MINDIAEI is presented in Algorithm 5, which has a total running time $\widetilde{O}(km)$.

5.2.2 Convex Hull Based Fast Iterative Algorithm . By definition, the resistance diameter of a graph is the largest value of resistance distances between all pairs of nodes in the graph. This enlightens us to minimize the resistance diameter by iteratively adding edges between the pair of nodes with the maximum resistance distance. Specifically, in each iteration step i, we only need to find the pair of nodes with the maximum resistance distance in the graph and establish a link between them. Actually, as shown in (4), the resistance distance is equivalent to Euclidean distance between two corresponding points in an Euclidean space. Then, Problem 1 can be approximately solved by finding the pair of points with the maximum distance in an Euclidean space at each iteration step. Based

Algorithm 6: MinDiaCH$(\mathcal{G}, Q, k, \epsilon)$

```
Input  : A connected graph $\mathcal{G} = (V, E)$, a candidate edge
         set $Q$, an integer $1 \le k \le |Q|$, a parameter $\epsilon$
Output : $P$ : A subset of $Q$ with $|P| = k$ edges
1  Initialize solution $P = \emptyset$
2  $d = \lceil 24 \log n / \epsilon^2 \rceil$, $\theta = \frac{\epsilon}{12}$
3  for $i = 1$ to $k$ do
4      $\tilde{X}_{d \times n} \leftarrow$ ApproxER$(\mathcal{G}, \epsilon)$
5      $S \leftarrow \{s_j \in \mathbb{R}^d | s_j = \tilde{X}_{[:,j]}, j = 1, 2, \ldots, n\}$
6      $\hat{S} \leftarrow$ ApproxCH$(S, \theta)$
7      Select $e_i$ s.t. $e_i \leftarrow \arg\max_{u,v \in \hat{S}, (u,v) \in Q \setminus P} ||\tilde{X}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$
8      Update solution $P \leftarrow P \cup \{e_i\}$
9      Update the graph $\mathcal{G} \leftarrow \mathcal{G}(\{e_i\})$
10 return $P$
```

on this idea, we propose a fast algorithm MinDiaCH for Problem 1, the pseudocode of which is illustrated in Algorithm 6.

The algorithm MinDiaCH is based on the ApproxCH algorithm in Lemma 4.5. For n points in a Euclidean space, the points that lie on the boundary of the convex hull of this point set are the farthest from one another. Using ApproxCH, we can obtain a set $\hat{S}$ of l nodes, all of which lie on the boundary of the convex hull. Our algorithm, MinDiaCH, selects the pair of nodes of $\hat{S}$ that are farthest apart at each step, which allows us to identify the pair of nodes in the original graph with the greatest resistance distance. With this approach, we achieve a fast and efficient solution to Problem 1. The total running time of MinDiaCH is $\widetilde{O}\left(k(m + nl)/\epsilon^2\right)$.

6 EXPERIMENTS

In this section, we present experimental results to evaluate the performance of the proposed FastRD algorithm for computing the resistance diameter, and MinDiaCH and MinDiaEi for minimizing it by adding a specified number of edges. The source code is publicly available on https://anonymous.4open.science/r/ERdiam-612E.

6.1 Experimental Setup

Datasets and Equipment. To evaluate the performance of our proposed approximation algorithms, we perform experiments on different realistic networks representatively selected from various domains, which are from Koblenz Network Collection [34] and Network Repository [53]. Table 1 reports the related information for the considered real networks. All our experiments are conducted on a Linux box with an Intel i7-7700K @ 4.2-GHz (4 Cores) and with 128-GB RAM. All algorithms are implemented in *Julia v1.0.3*, where the Laplace Solver is from [35].

6.2 Performance of Algorithm FastRD Computing for Resistance Diameter

6.2.1 Results on Realistic Networks. We now evaluate the performance of the FastRD algorithm in terms of efficiency and accuracy on real-world networks. For this purpose, we compare our algorithm FastRD with two methods: Exact and GEER [65] that is the state-of-the-art algorithm for querying resistance distances between node pairs. In [65], the error threshold for the algorithm GEER is set between 0.01 and 0.5. In our experiments, the error threshold of algorithm GEER is set to be 0.1.

We first evaluate the efficiency of the algorithm FastRD. In Table 1, we report the running times of Exact, GEER, and FastRD on several real-world networks. To objectively assess the execution time for the three algorithms on all considered networks, we enforced the program to run on a single thread. From Table 1, it is evident that both Exact and FastRD are faster than GEER. The main reasons are as follows: although GEER is adept at quickly querying the resistance distance between a single node pair, it requires running GEER $O(n^2)$ times to obtain the resistance diameter. Therefore, GEER is only suitable for small networks with fewer than 30,000 nodes. For larger networks, it cannot return results within 24 hours.

From Table 1, we can also observe that for small networks with fewer than 10,000 nodes, Exact is more efficient than FastRD. However, for larger networks and various approximation parameters ϵ, the computational time for FastRD is significantly smaller than that for Exact. For the last seven networks in Table 1, with node counts ranging from 10^6 to 10^7, we cannot run the Exact algorithm on our system due to memory and computational constraints. The Exact algorithm requires directly inverting the Laplacian matrix of the graph, which is computationally infeasible for large networks. In contrast, for these networks, we can approximately compute their resistance diameter by applying the FastRD algorithm, further demonstrating that FastRD is efficient and scalable to very large networks.

Except for the high efficiency, algorithm FastRD also provides a good approximation $\hat{R}(\mathcal{G})$ for the resistance diameter $R(\mathcal{G})$. To show the accuracy of FastRD, we compare the approximate values $\hat{R}(\mathcal{G})$ returned by FastRD with the rigorous results of $R(\mathcal{G})$ returned by Exact. In Table 1, we report the relative errors σ of algorithm FastRD, where σ is defined as $\sigma = (|R(\mathcal{G}) - \hat{R}(\mathcal{G})|/R(\mathcal{G}))$. From Table 1 we can see that the actual relative errors for all ϵ and all networks are very small, and are almost negligible for smaller ϵ. More interestingly, for all networks tested, σ are magnitudes smaller than the theoretical guarantee. Therefore, the approximation algorithm FastRD provides a very accurate result for resistance diameter in practice.

6.2.2 Results on Model Networks. For a general graph $\mathcal{G}$, to obtain an exact expression for the resistance diameter $R(\mathcal{G})$ is very difficult and even impossible. However, we can obtain closed-form solution to this quantity for some deterministic model networks, such as the hierarchical graphs [50], the Hanoi graphs [54], the Sierpiński gaskets [31], as well as and the Koch networks [73]. The detailed descriptions of the four model networks, along with their corresponding expressions for resistance diameter, are provided in the appendix.

To further demonstrate the performance of our algorithm FastRD approximating resistance diameter, we apply it to estimate the resistance diameter for the aforementioned four deterministic model networks: hierarchical graphs, Hanoi graphs, Sierpiński Gasket graphs and Koch networks. In Table 2, we report the exact resistance diameter $R(\mathcal{G})$, the approximation $\hat{R}(\mathcal{G})$, relative error

Table 1: The running time (seconds, s) of Exact, GEER and FastRD with various ϵ, as well the relative error σ ($\times 10^{-2}$) on various networks. For each network $\mathcal{G}$, we indicate the number of nodes n, the number of edges m and the resistance diameter $R(\mathcal{G})$.

Network	n	m	$R(\mathcal{G})$	Running time (s)				Relative error σ		
				Exact	GEER	FastRD		GEER	FastRD	
						0.2	0.1		0.2	0.1
EmailUN	1,133	5,451	4.278	0.418	129.311	4.273	7.215	0.02	0.82	0.18
USGrid	4,941	6,594	18.283	9.357	1412.265	10.014	16.528	0.03	0.17	0.02
Government	7,057	89,429	6.207	20.331	5433.781	13.792	31.445	0.08	0.66	0.16
Hep-th	8,361	15,751	12.154	13.112	8834.422	30.776	94.534	0.11	0.26	0.14
Athletes	13,866	86,811	6.872	89.722	31521.387	32.452	66.587	0.55	0.87	0.56
Musae-facebook	22,470	170,823	8.956	261.931	74842.871	113.449	150.819	0.07	0.88	0.02
New-sites	27,917	205,964	10.016	426.714	–	96.414	159.973	–	0.34	0.07
Musae-git	37,700	289,003	7.522	890.925	–	182.239	220.131	–	0.91	0.26
HU	47,538	222,887	8.018	2024.661	–	219.461	370.527	–	0.49	0.23
HR	54,573	498,202	6.446	2334.059	–	209.618	431.59	–	0.74	0.55
Epinions	75,877	508,836	10.868	5555.885	–	318.196	416.280	–	1.05	0.87
Slashdot	77,360	828,161	5.009	5854.217	–	386.418	574.755	–	0.77	0.61
Delicious*	536,108	1,365,961	9.524	–	–	1095.641	2062.206	–	–	–
FourSquare*	639,014	3,214,986	3.018	–	–	1236.412	2667.382	–	–	–
Lastfm-song*	1,085,612	19,150,868	4.005	–	–	3937.847	5887.932	–	–	–
Edit*	1,347,094	5,276,371	6.31	–	–	2343.134	3811.039	–	–	–
Wikipedia-growth*	1,870,709	39,953,145	6.77	–	–	5441.601	6891.479	–	–	–
Flixster*	2,523,386	7,918,801	8.31	–	–	3617.261	8587.502	–	–	–

Table 2: Exact resistance diameters $R(\mathcal{G})$, their approximations $\hat{R}(\mathcal{G})$, relative error $\rho = (|R(\mathcal{G}) - \hat{R}(\mathcal{G})|/R(\mathcal{G}))$, and running time (seconds, s) for $\hat{R}(\mathcal{G})$ on four different networks.

Network	Vertices	Edges	$R(\mathcal{G})$	$\hat{R}(\mathcal{G})$	ρ	Time
$\mathcal{H}(13)$	1,594,323	2,391,483	16.667	16.498	0.0101	1472
$\mathcal{T}(12)$	1,594,323	2,391,483	764.656	766.947	0.0029	1521
$\mathcal{S}(13)$	2,391,486	4,782,969	510.437	514.247	0.0075	2303
$\mathcal{K}(10)$	2,097,153	3,145,728	14	13.89	0.0078	2256

$\rho = (|R(\mathcal{G}) - \hat{R}(\mathcal{G})|/R(\mathcal{G}))$, and running time (seconds, s) for $\hat{R}(\mathcal{G})$ on four deterministic graphs. The corresponding approximation $\hat{R}(\mathcal{G})$ is obtained via our algorithm FastRD with $\epsilon = 0.1$. Table 2 shows that FastRD works effectively and efficiently for all the four studied deterministic networks. This again demonstrates the superiority of the algorithm FastRD in approximating the resistance diameter of huge networks with millions of nodes.

6.3 Performance of Algorithms for Minimizing Resistance Diameter

6.3.1 Baseline Methods. To evaluate the performance of our proposed algorithms, Simple, MinDiaEi, and MinDiaCH, we compared them against six baseline edge selection methods: Optimum, Random, Closeness, PageRank, Diameter, and Wspd, with details provided in the appendix.

6.3.2 Effectiveness. We first study the effectiveness of our algorithms Simple, MinDiaEi and MinDiaCH, by comparing them with Optimum, Random and Wspd. We execute experiments on four small real networks: Kangaroo with 17 nodes and 91 edges, Rhesus

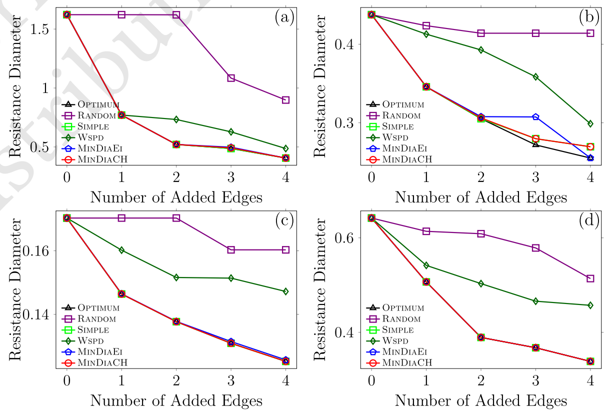

Figure 1: Resistance diameters of the augmented graphs returned by our algorithms, as well as three baseline strategies on four networks: (a) Kangaroo, (b) Rhesus, (c) Cloister, and (d) Tribes.

with 16 nodes and 111 edges, Cloister with 18 nodes and 189 edges and Tribes with 16 nodes and 58 edges. Their small size allows us to compute the optimal set of added edges. Since computing the optimal solution requires exponential time, we only consider $k = 0, 1, 2, 3, 4$. The results are shown in Figure 1, which shows that the resistance diameters of the augmented graphs returned by our three greedy algorithms and the optimum solutions are almost the same, all of which are much better than those returned by the Random and Wspd schemes.

In order to further demonstrate the effectiveness MinDiaEi and MinDiaCH, we also compare their returned results with five baseline schemes: Random, Diameter, Closeness, PageRank, and

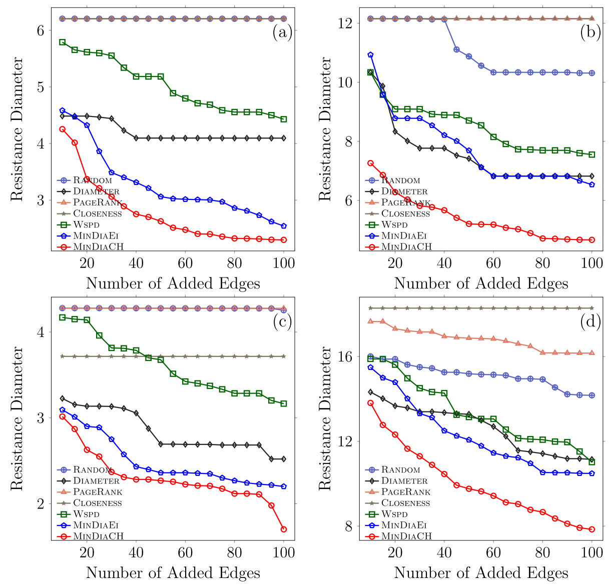

Figure 2: Resistance diameters for the augmented graphs returned by our two algorithms, and five baseline heuristics on four real netowrks: (a) Government, (b) Hep-th, (c) EmailUN, and (d) USGrid.

Wspd, on four large real networks. For each network, we calculate the resistance diameter in the original graph. Then we decrease the resistance diameter by adding up to $k = 1, 2, \ldots, 100$ new edges, applying our greedy algorithms and the five baseline strategies of edge addition. After adding each edge by different methods, we compute and record the resistance diameters. The results are shown in Figure 2, which indicates that for each network, our two greedy algorithms outperform the baseline strategies.

Finally, we execute experiments on four much larger networks to display the effectiveness of MinDiaEi and MinDiaCH. For these networks, we cannot run baselines Diameter, Closeness, PageRank, due to their high complexity, but only run Random and Wspd. The results are reported in Figure 3, which shows that MinDiaEi and MinDiaCH return significantly better results than Random and Wspd.

6.3.3 Efficiency. As demonstrated above, in comparison with the baseline strategies of edge addition, both of our algorithms MinDiaCH and MinDiaEi exhibit good effectiveness. Here we study the efficiency of our algorithms MinDiaCH and MinDiaEi. For this purpose, we run these two algorithms on four large-scale networks, with the largest one having over a million nodes. For each network, we select $k = 100$ and record the running time of both algorithms in Table 3. The results show that both MinDiaCH and MinDiaEi are efficient, and are scalable to networks with millions of nodes. However, there is some difference for the performance between MinDiaCH and MinDiaEi. As shown in Table 3, MinDiaEi runs faster than MinDiaCH. For example, MinDiaCH does not terminate in one day for the network Edit, while MinDiaEi outputs the solution within 2 hours. Although MinDiaEi is more efficient than MinDiaCH, Figures 2 and 3 illustrate that MinDiaCH is more effective than MinDiaEi.

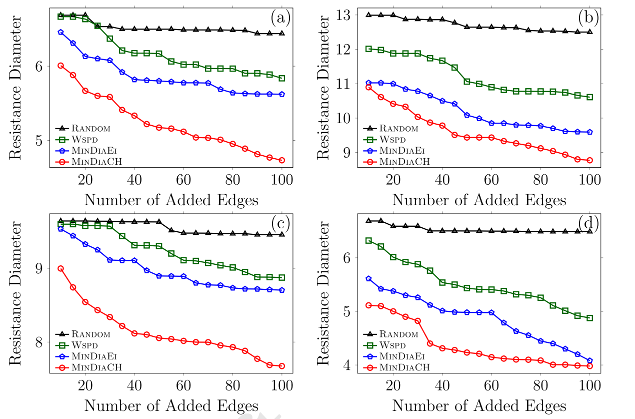

Figure 3: Resistance diameters for the augmented graphs returned by our algorithms, and two baseline schemes on four large networks: (a) HR, (b) Epinions, (c) Delicious, and (d) Edit.

Table 3: The running time (seconds, s) of MinDiaCH and MinDiaEi on some large-scale networks.

Network	Vertices	Edges	Running time (s)	
			MinDiaCH	MinDiaEi
HR	54,573	498,202	9700	335
Epinions	75,877	508,836	9800	490
Delicious	536,108	1,365,961	71080	1491
Edit	1,347,094	5,276,371	168900	4930

7 CONCLUSION

In this paper, we present FastRD, a fast approximation algorithm for efficiently computing the resistance diameter of a graph with a guaranteed error. Direct computation of resistance diameter is computationally expensive, requiring matrix inversion and pairwise distance calculations for all nodes. To address this, FastRD employs the Johnson-Lindenstrauss Lemma, Laplacian solvers, and convex hull techniques, significantly reducing the computational complexity by focusing on estimating resistance distances for a smaller subset of node pairs. This allows the algorithm to scale effectively to large graphs with millions of nodes.

We also introduce the problem of minimizing the resistance diameter by adding k edges to the graph. We proved that the problem is NP-hard and that its objective function is non-supermodular but monotone. To solve this, we propose three heuristic algorithms: Simple, MinDiaEi, and MinDiaCH, with varying trade-offs between speed and effectiveness. Extensive experiments demonstrate that all algorithms are efficient and scalable, with MinDiaEi being faster and MinDiaCH offering better results in terms of resistance diameter reduction. Future work will focus on extending these techniques to optimize other graph metrics, such as biharmonic distance.

REFERENCES

[1] Florian Adriaens and Aristides Gionis. 2022. Diameter minimization by shortcutting with degree constraints. In *2022 IEEE International Conference on Data Mining*. IEEE, 843–848.

[2] Ali Aghdaei and Zhuo Feng. 2022. HyperEF: Spectral Hypergraph Coarsening by Effective-Resistance Clustering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*. 1–9.

[3] Vedat Levi Alev, Nima Anari, Lap Chi Lau, and Shayan Oveis Gharan. 2018. Graph clustering using effective resistance. 94 (2018), 41.

[4] Nima Anari and Shayan Oveis Gharan. 2015. Effective-Resistance-Reducing Flows, Spectrally Thin Trees, and Asymmetric TSP. In *Proc. 56th Annu. IEEE Symp. Found. Comput. Sci.* Berkeley, CA, USA, 20–39.

[5] Pranjal Awasthi, Bahman Kalantari, and Yikai Zhang. 2020. Robust vertex enumeration for convex hulls in high dimensions. *Ann. Oper. Res.* 295, 1 (2020), 37–73.

[6] Joshua Batson, Daniel A Spielman, Nikhil Srivastava, and Shang-Hua Teng. 2013. Spectral sparsification of graphs: Theory and algorithms. *Commun. ACM* 56, 8 (2013), 87–94.

[7] Régis Behmo, Nikos Paragios, and Véronique Prinet. 2008. Graph commute times for image representation. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.

[8] David Belius and Nicola Kistler. 2017. The subleading order of two dimensional cover times. *Probability Theory and Related Fields* 167, 1-2 (2017), 461–552.

[9] Adi Ben-Israel and Thomas N. E Greville. 1974. *Generalized inverses: Theory and applications.* J. Wiley. 406–413 pages.

[10] Ulrik Brandes and Daniel Fleischer. 2005. Centrality measures based on current flow. In *Proceedings of Annual Symposium on Theoretical Aspects of Computer Science*, Vol. 3404. Springer, 533–544.

[11] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Trans. Knowl. Data. Eng.* 30, 9 (2018), 1616–1637.

[12] Paul B Callahan and S Rao Kosaraju. 1995. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM* 42, 1 (1995), 67–90.

[13] Pak Hay Chan, Lap Chi Lau, Aaron Schild, Sam Chiu-wai Wong, and Hong Zhou. 2022. Network design for $s-t$ effective resistance. *ACM Transactions on Algorithms* 18, 3 (2022), 1–45.

[14] Bernard Chazelle. 1993. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry* 10, 4 (1993), 377–409.

[15] James Cheng, Silu Huang, Huanhuan Wu, and Ada Wai-Chee Fu. 2013. TF-Label: A topological-folding labeling scheme for reachability querying in a large graph. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. 193–204.

[16] Paul Christiano, Jonathan A. Kelner, Aleksander Madry, Daniel A. Spielman, and Shang-Hua Teng. 2011. Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs. In *Proc. 43rd Annu. ACM Symp. Theory Comput.* San Jose, CA, USA, 273–282.

[17] Alessio Conte, Tiziano De Matteis, Daniele De Sensi, Roberto Grossi, Andrea Marino, and Luca Versari. 2018. D2K: scalable community detection in massive networks via small-diameter k-plexes. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1272–1281.

[18] Erik D Demaine and Morteza Zadimoghaddam. 2010. Minimizing the diameter of a network using shortcut edges. In *Algorithm Theory-SWAT 2010: 12th Scandinavian Symposium and Workshops on Algorithm Theory, Bergen, Norway, June 21-23, 2010. Proceedings 12*. Springer, 420–431.

[19] Florian Dörfler, John W Simpson-Porco, and Francesco Bullo. 2018. Electrical Networks and Algebraic Graph Theory: Models, Properties, and Applications. *Proc. IEEE* 106, 5 (2018), 977–1005.

[20] Peter G Doyle and J Laurie Snell. 1984. *Random walks and electric networks*. Vol. 22. American Mathematical Soc.

[21] Francois Fouss, Alain Pirotte, Jean-michel Renders, and Marco Saerens. 2007. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Trans. Knowl. Data. Eng.* 19, 3 (Jan. 2007), 355–369.

[22] Fabrizio Frati, Serge Gaspers, Joachim Gudmundsson, and Luke Mathieson. 2015. Augmenting graphs to minimize the diameter. *Algorithmica* 72 (2015), 995–1010.

[23] Scott Freitas, Diyi Yang, Srijan Kumar, Hanghang Tong, and Duen Horng Chau. 2023. Graph vulnerability and robustness: A survey. *IEEE Transactions on Knowledge and Data Engineering* 35, 6 (2023), 5915–5934.

[24] Michael R Garey, David S Johnson, and Larry Stockmeyer. 1974. Some simplified NP-complete problems. In *Proceedings of the sixth Annual ACM Symposium on Theory of Computing*. 47–63.

[25] Severino V Gervacio. 2016. Resistance distance in complete n-partite graphs. *Discrete Applied Mathematics* 203 (2016), 53–61.

[26] Arpita Ghosh and Stephen Boyd. 2006. Growing well-connected graphs. In *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE, 6605–6611.

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

[28] Arvind Giridhar and Praveen R Kumar. 2006. Distributed clock synchronization over wireless networks: Algorithms and analysis. In *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE, 4915–4920.

[29] Takanori Hayashi, Takuya Akiba, and Yuichi Yoshida. 2016. Efficient algorithms for spanning tree centrality. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence*. 3733–3739.

[30] Guixian Huang, Weihua He, and Yuanyao Tan. 2019. Theoretical and computational methods to minimize Kirchhoff index of graphs with a given edge k-partiteness. *Appl. Math. Comput.* 341 (2019), 348–357.

[31] Zhuozhuo Jiang and Weigen Yan. 2018. Some two-point resistances of the Sierpinski gasket network. *Journal of Statistical Physics* 172 (2018), 824–832.

[32] William B Johnson and Joram Lindenstrauss. 1984. Extensions of Lipschitz mappings into a Hilbert space. *Contemp. Math.* 26 (1984), 189–206.

[33] Douglas J Klein and Milan Randić. 1993. Resistance distance. *Journal of Mathematical Chemistry* 12 (1993), 81–95.

[34] Jérôme Kunegis. 2013. Konect: The koblenz network collection. In *Proceedings of the 22nd International World Wide Web Conference*. ACM, 1343–1350.

[35] Rasmus Kyng and Sushant Sachdeva. 2016. Approximate gaussian elimination for Laplacians-fast, sparse, and simple. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science*. IEEE, 573–582.

[36] Huan Li, Stacy Patterson, Yuhao Yi, and Zhongzhi Zhang. 2020. Maximizing the number of spanning trees in a connected graph. *IEEE Trans. Inf. Theory* 66, 2 (2020), 1248–1260.

[37] Huan Li, Richard Peng, Yuhao Shan, Lirenand Yi, and Zhongzhi Zhang. 2019. Current Flow Group Closeness Centrality for Complex Networks?. In *Proc. World Wide Web Conf.* San Francisco, USA, 961–971.

[38] Ye Li, Leong Hou U, Man Lung Yiu, and Ngai Meng Kou. 2017. An experimental study on hub labeling based shortest path algorithms. *Proceedings of the VLDB Endowment* 11, 4 (2017), 445–457.

[39] Meihao Liao, Rong-Hua Li, Qiangqiang Dai, Hongyang Chen, Hongchao Qin, and Guoren Wang. 2023. Efficient Resistance Distance Computation: The Power of Landmark-based Approaches. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–27.

[40] Qing Liu, Minjun Zhao, Xin Huang, Jianliang Xu, and Yunjun Gao. 2020. Truss-based community search over large directed graphs. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 2183–2197.

[41] Aleksander Madry, Damian Straszak, and Jakub Tarnawski. 2015. Fast Generation of Random Spanning Trees and the Effective Resistance Metric. In *Proc. 26th Annu. ACM-SIAM Symp. Discrete Algorithms.* San Diego, CA, USA, 2019–2036.

[42] Víctor Martínez, Fernando Berzal, and Juan-Carlos Cubero. 2016. A survey of link prediction in complex networks. *ACM Comput. Surv.* 49, 4 (2016), 1–33.

[43] Charalampos Mavroforakis, Richard Garcia-Lebron, Ioannis Koutis, and Evimaria Terzi. 2015. Spanning edge centrality: Large-scale computation and applications. In *Proceedings of the 24th International Conference on World Wide Web*. 732–742.

[44] Songsong Mo, Zhifeng Bao, Ping Zhang, and Zhiyong Peng. 2020. Towards an efficient weighted random walk domination. *Proceedings of the VLDB Endowment* 14, 4 (2020), 560–572.

[45] Damon Mosk-Aoyama. 2008. Maximum algebraic connectivity augmentation is NP-hard. *Operations Research Letters* 36, 6 (2008), 677–679.

[46] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Mathematical Programming* 14 (1978), 265–294.

[47] Kobbi Nissim, Uri Stemmer, and Salil Vadhan. 2016. Locating a small cluster privately. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 413–427.

[48] Pan Peng, Daniel Lopatta, Yuichi Yoshida, and Gramoz Goranci. 2021. Local algorithms for estimating effective resistance. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1329–1338.

[49] Franco P Preparata and Michael Ian Shamos. 1985. Computational geometry: An introduction. *Texts and Monographs in Computer Science* (1985).

[50] Yi Qi, Yuhao Yi, and Zhongzhi Zhang. 2019. Topological and spectral properties of small-world hierarchical graphs. *Comput. J.* 62, 5 (2019), 769–784.

[51] Yi Qi, Zhongzhi Zhang, Yuhao Yi, and Huan Li. 2019. Consensus in self-similar hierarchical graphs and Sierpiński graphs: Convergence speed, delay robustness, and coherence. *IEEE Trans. Cybern.* 49, 2 (2019), 592–603.

[52] Jiezhong Qiu, Laxman Dhulipala, Jie Tang, Richard Peng, and Chi Wang. 2021. Lightne: A lightweight graph processing system for network embedding. In *Proceedings of the 2021 International Conference on Management of Data*. 2281–2289.

[53] Ryan Rossi and Nesreen Ahmed. 2015. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI, 4292–4293.

[54] Muhammad Shoaib Sardar, Xiang-Feng Pan, and Yun-Xiang Li. 2020. Some two-vertex resistances of the three-towers Hanoi graph formed by a fractal graph. *Journal of Statistical Physics* 181, 1 (2020), 116–131.

[55] Liren Shan, Yuhao Yi, and Zhongzhi Zhang. 2018. Improving information centrality of a node in complex networks by adding edges. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3535–3541.
[56] Jieming Shi, Nikos Mamoulis, Dingming Wu, and David W Cheung. 2014. Density-based place clustering in geo-social networks. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 99–110.
[57] Daniel A Spielman and Nikhil Srivastava. 2008. Graph sparsification by effective resistances. In *Proceedings of the fortieth Annual ACM Symposium on Theory of Computing*. 563–568.
[58] Kumar Sricharan and Kamalika Das. 2014. Localizing anomalous changes in time-evolving graphs. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. 1347–1358.
[59] Ali Tizghadam and Alberto Leon-Garcia. 2010. Autonomic traffic engineering for network robustness. *IEEE J. Sel. Areas Commun.* 28, 1 (Jan. 2010).
[60] Leo Torres, Kevin S Chan, Hanghang Tong, and Tina Eliassi-Rad. 2021. Nonbacktracking eigenvalues under node removal: X-centrality and targeted immunization. *SIAM Journal on Mathematics of Data Science* 3, 2 (2021), 656–675.
[61] Ye Wang, Qing Wang, Henning Koehler, and Yu Lin. 2021. Query-by-sketch: Scaling shortest path graph queries on very large networks. In *Proceedings of the 2021 International Conference on Management of Data*. 1946–1958.
[62] Hao Wei, Jeffrey Xu Yu, Can Lu, and Xuemin Lin. 2016. Speedup graph processing by graph ordering. In *Proceedings of the 2016 International Conference on Management of Data*. 1813–1828.
[63] Wenjun Xiao and Ivan Gutman. 2003. Resistance distance and Laplacian spectrum. *Theoretical Chemistry Accounts* 110, 4 (2003), 284–289.
[64] Wanyue Xu and Zhongzhi Zhang. 2023. Optimal scale-free small-world graphs with minimum scaling of cover time. *ACM Transactions on Knowledge Discovery from Data* 17, 7 (2023), 93.
[65] Renchi Yang and Jing Tang. 2023. Efficient estimation of pairwise effective resistance. *Proceedings of the ACM on Management of Data* 1, 1 (2023), 1–27.
[66] Pranali Yawalkar and Sayan Ranu. 2019. Route recommendations on road networks for arbitrary user preference functions. In *2019 IEEE 35th International Conference on Data Engineering*. IEEE, 602–613.
[67] Yuhao Yi, Bingjia Yang, Zuobai Zhang, Zhongzhi Zhang, and Stacy Patterson. 2022. Biharmonic distance-based performance metric for second-order noisy consensus networks. *IEEE Trans. Inf. Theory* 68, 2 (2022), 1220–1236.
[68] Yuhao Yi, Zhongzhi Zhang, and Stacy Patterson. 2020. Scale-free loopy structure is resistant to noise in consensus dynamics in complex networks. *IEEE Trans. Cybern.* 50, 1 (2020), 190–200.
[69] Yuhao Yi, Zhongzhi Zhang, Liren Shan, and Guanrong Chen. 2017. Robustness of first-and second-order consensus algorithms for a noisy scale-free small-world Koch network. *IEEE Trans. Control Syst. Technol.* 25, 1 (2017), 342–350.
[70] Yikai Zhang and Jeffrey Xu Yu. 2020. Hub labeling for shortest path counting. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1813–1828.
[71] Zhongzhi Zhang and Shuyang Gao. 2011. Scaling of mean first-passage time as efficiency measure of nodes sending information on scale-free Koch networks. *The European Physical Journal B* 80 (2011), 209–216.
[72] Zhongzhi Zhang, Shunqi Wu, Mingyun Li, and Francesc Comellas. 2016. The number and degree distribution of spanning trees in the Tower of Hanoi graph. *Theoret. Comput. Sci.* 609 (2016), 443–455.
[73] Zhongzhi Zhang, Shuigeng Zhou, Wenlei Xie, Lichao Chen, Yuan Lin, and Jihong Guan. 2009. Standard random walks and trapping on the Koch network with scale-free behavior and small-world effect. *Physical Review E* 79, 6 (2009), 061113.
[74] Xiaotian Zhou and Zhongzhi Zhang. 2021. Maximizing influence of leaders in social networks. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2400–2408.
[75] David Zuckerman. 1992. A technique for lower bounding the cover time. *SIAM Journal on Discrete Mathematics* 5, 1 (1992), 81–87.

A PROOF OF LEMMA 4.6

Proof. Let $u' \in C(\hat{S})$ and $v' \in C(\hat{S})$ be two points, which are the closest ones to u and v, respectively. By Lemma 4.5 and the fact $D(S) = D(\bar{S})$, we have $d(u, u') \leq \theta D(\bar{S})$ and $d(v, v') \leq \theta D(\bar{S})$. Considering $d(x, y) = D(\hat{S})$ and the condition that u' and v' belong to the convex hull $C(\hat{S})$, we have $d(x, y) \geq d(u', v')$. Making use of the triangle inequality twice, it follows that $d(x, y) \geq d(u', v') \geq \left(d(u, v) - 2\theta D(\bar{S})\right)$. Note that $d(u, v) = D(\bar{S})$ and $\theta = \frac{\epsilon}{12}$, we obtain $D(\bar{S}) \geq d(x, y) \geq (1 - \frac{\epsilon}{6})D(\bar{S})$, which directly leads to (6). □

B PROOF OF LEMMA 4.7

Proof. Let $D(\bar{S})^2 = \max_{u,v\in\bar{S}} d(u, v)^2$. By Lemma 4.6, we have $D(\bar{S}) \geq D(\hat{S}) \geq (1 - \frac{\epsilon}{6})D(\bar{S})$. Then we obtain

$$\begin{aligned} D(\bar{S})^2 \geq D(\hat{S})^2 &\geq (1 - \frac{\epsilon}{6})^2 D(\bar{S})^2 \\ &= (1 - \frac{\epsilon}{3} + \frac{\epsilon^2}{36})D(\bar{S})^2 \\ &\geq (1 - \frac{\epsilon}{3})D(\bar{S})^2 \end{aligned}$$

On the other hand, we have $\bar{R}(\mathcal{G}) = D(S)^2 = D(\bar{S})^2$. Combining the above-obtained results, we have $\bar{R}(\mathcal{G}) \geq \hat{R}(\mathcal{G}) \geq (1 - \frac{\epsilon}{3})\bar{R}(\mathcal{G})$ which leads to (7). □

C PROOF OF THEOREM 4.8

Proof. We first analyze the time complexity of algorithm FastRD$(\mathcal{G}, \epsilon)$, which includes two main operations: constructing matrix $\tilde{X}$ and determining set $\hat{S}$. The construction of $\tilde{X}$ takes $\widetilde{O}\left(m/\epsilon^2\right)$ time, while finding the set $\hat{S}$ of points needs $\widetilde{O}(nl/\epsilon^2)$ time. Thus, the total running time of algorithm FastRD is $\widetilde{O}\left((m + nl)/\epsilon^2\right)$.

We proceed to prove the correctness of the approximation error. Let $\bar{R}(\mathcal{G}) = \max_{u,v\in V} ||\tilde{X}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$, and let $\hat{R}(\mathcal{G}) = \max_{u,v\in\hat{S}} ||\tilde{X}(\boldsymbol{e}_u - \boldsymbol{e}_v)||_2^2$. Then, we have $\bar{R}(\mathcal{G}) \overset{\epsilon}{\approx} R(\mathcal{G})$ and $\hat{R}(\mathcal{G}) \overset{\epsilon/3}{\approx} \bar{R}(\mathcal{G})$, both of which result in $(1 - \epsilon)R(\mathcal{G}) \leq \hat{R}(\mathcal{G}) \leq (1 + \epsilon)R(\mathcal{G})$, completing the proof. □

D HARDNESS OF RESISTANCE DIAMETER MINIMIZATION PROBLEM

The objective function of Problem 1 is not explicit, suggesting that Problem 1 seems to be difficult. In this subsection, we confirm this intuition by proving that Problem 1 is NP-hard. We will give a reduction from the 3-colorability problem, which has been shown to be NP-hard [24]. First, we introduce a minimum resistance diameter augmentation problem.

Problem 2. *(Minimum Resistance Diameter Augmentation Problem) Given an undirected, connected, simple graph $\mathcal{G} = (V, E)$, a non-negative integer k and a non-negative threshold R_0, is there a subset $P \subseteq Q$ of size $|P| \leq k$ with $Q = (V \times V)\backslash E$, such that the graph $\mathcal{H} = (V, E \cup P)$ satisfies $R(\mathcal{H}) \leq R_0$?*

We now show that Problem 2 is in NP. For any given graph $\mathcal{G}$ and the set P of added edges, the correctness of a given solution for Problem 2 can be verified by computing the resistance distances for all $O(n^2)$ pairs of nodes, which can be obtained by inverting an associated matrix in $O(n^3)$ time. Then using (3) to get the resistance diameter and comparing the outcome with the given threshold R_0 verifies the solution. Therefore, the minimum resistance diameter augmentation problem is in NP.

In fact, Problem 2 is the decision version of the following optimization problem: Given an undirected, connected, simple graph $\mathcal{G} = (V, E)$ and a non-negative threshold R_0, find a set of currently non-existent edges of minimum size to add to $\mathcal{G}$ such that the resistance diameter of the resultant augmented graph is at most R_0. The main work of this section is to prove that Problem 2 is NP-hard, which immediately implies that the corresponding optimization problem is also NP-hard. Thus, the problem of creating a fixed number of edges to a graph to minimize the resistance diameter of the augmented graph is also NP-hard.

Our proof of the NP-hardness for Problem 2 is inspired by the proof of the NP-hardness of the maximum algebraic connectivity augmentation problem [45]. For a graph $\mathcal{G} = (V, E)$ with n nodes and m edges, we construct a new $\mathcal{G}' = (V', E')$ as in [45]. Graph $\mathcal{G}' = (V'E')$ consists of three disjoint copies $\mathcal{G}_0, \mathcal{G}_1$ and $\mathcal{G}_2$ of graph $\mathcal{G}$. For every node $v \in V$, there is a corresponding node $v_i \in \mathcal{G}_i$ ($i = 0, 1, 2$); and for each edge $(u, v) \in E$, there is a corresponding edge $(u_i, v_i) \in \mathcal{G}_i$ ($i = 0, 1, 2$). By construction the graph $\mathcal{G}'$ has $3n$ vertices and $3m$ edges. We now consider the minimum resistance diameter augmentation problem on $\mathcal{G}'$ with $k = 3n^2 - 3m$, such that the augmented graph $\mathcal{H}$ has at most $3n^2$ edges and $R_0 = \frac{1}{n}$.

For our proof, we introduce a class of graphs. Let $\mathcal{K}_{n,n,n}$ denote the complete tripartite graph, consisting of three disjoint groups of nodes, with each having exact n nodes. In $\mathcal{K}_{n,n,n}$, there is no edge linking any two nodes within the same group, while every node in one group is adjacent to every node in the other two groups. Thus, the graph $\mathcal{K}_{n,n,n}$ has $3n$ vertices and $3n^2$ edges. Using the previous result in [25], it is straightforward to obtain the resistance diameter for $\mathcal{K}_{n,n,n}$.

Lemma D.1. *For the complete tripartite graph $\mathcal{K}_{n,n,n}$ with $3n$ nodes and $3n^2$ edges, let $R(\mathcal{K}_{n,n,n})$ be its resistance diameter. Then, $R(\mathcal{K}_{n,n,n}) = \frac{1}{n}$.*

Proof. In the complete tripartite graph $\mathcal{K}_{n,n,n}$, the resistance distance $r(i, j)$ between any pair of nodes i and j can be determined explicitly [25]. If i and j are in the same group, $r(i, j) = \frac{1}{n}$; If i and j are in different groups, $r(i, j) = \frac{3n-1}{3n^2}$. Thus, according to (3), we obtain $R(\mathcal{K}_{n,n,n}) = \max\{\frac{1}{n}, \frac{3n-1}{3n^2}\} = \frac{1}{n}$. □

In order to prove that the minimum resistance diameter augmentation problem can be reduced to the 3-colorability problem, we need the following lemmas.

Lemma D.2. *[45] There exists a subset $P \subseteq Q$ of size $|P| \leq k$, where $Q = (V \times V)\backslash E$, such that $\mathcal{H} = (V', E' \cup P)$ is isomorphic to $\mathcal{K}_{n,n,n}$ if and only if $\mathcal{G}$ is 3-colorable.*

Lemma D.3. *[45] For any graph $\mathcal{H} = (V, E)$ with $|V| = 3n$ nodes and $|E| \leq 3n^2$ edges for $n > 1$ satisfies $\lambda_2(\mathcal{H}) \geq 2n$ if and only if $\mathcal{H}$ is isomorphic to $\mathcal{K}_{n,n,n}$.*

Lemma D.4. *[63] For a connected graph $\mathcal{G}$ with Laplacian matrix $\boldsymbol{L}$, let $R(\mathcal{G})$ be the resistance diameter of $\mathcal{G}$, and let λ_2 be the smallest non-zero eigenvalue of $\boldsymbol{L}$. Then, $R(\mathcal{G}) \leq \frac{2}{\lambda_2}$.*

Lemma D.5. *For any graph $\mathcal{H} = (V, E)$ with $|V| = 3n$ nodes and $|E| \leq 3n^2$ edges for $n > 1$ satisfies its resistance diameter $R(\mathcal{H}) \leq \frac{1}{n}$ if and only if $\mathcal{H}$ is isomorphic to $\mathcal{K}_{n,n,n}$.*

Proof. Based on Lemma D.4, we have $R(\mathcal{H}) \leq \frac{2}{\lambda_2(\mathcal{H})}$. Using Lemma D.1 and Lemma D.3, if $\mathcal{H}$ is isomorphic to $\mathcal{K}_{n,n,n}$, we have $R(\mathcal{H}) = R(\mathcal{K}_{n,n,n}) = \frac{1}{n}$. On the other hand, Lemma D.3 indicates that $\lambda_2(\mathcal{H}) \geq 2n$, which, together with Lemma D.4, leads to $R(\mathcal{H}) \leq \frac{2}{\lambda_2(\mathcal{H})} \leq \frac{1}{n}$. Combining the above results, $R(\mathcal{H}) = \frac{1}{n}$ if $\mathcal{H}$ is isomorphic to $\mathcal{K}_{n,n,n}$, and $R(\mathcal{H}) > \frac{1}{n}$ otherwise. □

Lemma D.2 and Lemma D.5 directly lead to our main result of this section.

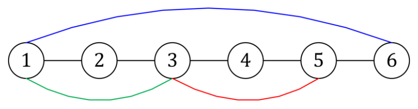

Figure 4: A line graph with 6 nodes and 5 edges. The colored lines represent newly added edges.

Theorem D.6. *The minimum resistance diameter augmentation problem is NP-hard.*

E NON-SUPERMODULAR PROPERTY OF THE OBJECTIVE FUNCTION

Two typical concepts related to an optimization problem with a cardinality constrain are monotone non-decreasing and supermodular set functions.

Definition E.1. *(Monotonicity) A set function $f : 2^Q \to \mathbb{R}$ is monotone non-decreasing if $f(S) \leq f(T)$ holds for all $S \subseteq T \subseteq Q$*

Definition E.2. *(Supermodularity) A set function $f : 2^Q \to \mathbb{R}$ is supermodular if*

$$f(S) - f(S \cup \{e\}) \geq f(T) - f(T \cup \{e\})$$

holds for all $S \subseteq T \subseteq V$ and $e \in Q$.

For a combinatorial optimization problem, when its objective function is monotone and supermodular, a simple greedy algorithm by selecting one element with the maximum marginal benefit in each iteration yields a solution with $(1 - e^{-1})$ approximation ratio [46]. However, there are still many combinatorial optimization problems whose objective functions are not supermodular. For these problems, a heuristic cannot guarantee a $(1 - e^{-1})$ approximation solution. Unfortunately, Problem 1 belongs to this problem class. Although the objective function for Problem 1 is monotone decreasing, next we show that it is non-supermodular. To this end, we give an example of the path graph $\mathcal{G}$ with 6 nodes and 5 edges in Figure 4. Let set $A = \{(1, 6)\}$, set $B = \{(1, 6), (1, 3)\}$ and edge $e = (3, 5)$. Simple computation leads to

$$\begin{aligned} R(\mathcal{G}(A)) &= 1.5, \qquad & R(\mathcal{G}(A \cup \{e\})) &= 1.5, \\ R(\mathcal{G}(B)) &= 1.5, \qquad & R(\mathcal{G}(B \cup \{e\})) &= 1.2, \end{aligned}$$

which means

$$R(\mathcal{G}(A)) - R(\mathcal{G}(A \cup \{e\})) < R(\mathcal{G}(B)) - R(\mathcal{G}(B \cup \{e\})),$$

violating the definition of supermodularity. Therefore, the objective function of Problem 1 is non-supermodular.

F MODEL NETWORKS AND THEIR RESISTANCE DIAMETERS

Hierarchical Graphs. Hierarchical graphs are constructed iteratively [50]. Starting with the complete graph $\mathcal{K}_3$ for $g = 1$, for $g \geq 2$, $\mathcal{H}(g)$ is created by generating three copies of $\mathcal{H}(g-1)$, then identifying the three vertices of a new complete graph $\mathcal{K}_3$ with the three hub nodes of each copy. Figure 5(a) illustrates the hierarchical graph $\mathcal{H}(3)$. In the hierarchical graph $\mathcal{H}(g)$, there are 3^g nodes

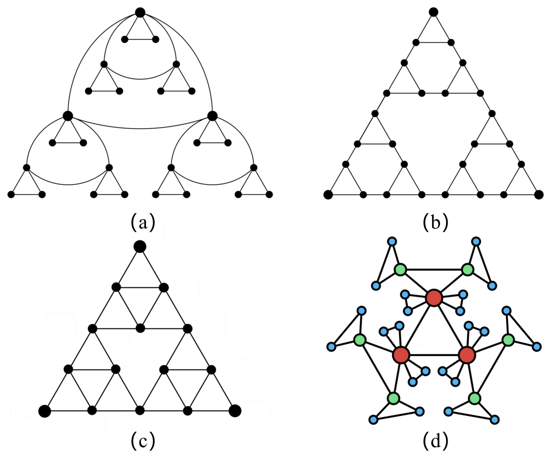

Figure 5: Illustration of the first several iterations of four deterministic graphs: (a) the hierarchical graph $\mathcal{H}(3)$, (b) the Towers of Hanoi graph $\mathcal{T}(3)$, (c) the Sierpiński gasket $\mathcal{S}(2)$, and (d) the Koch network $\mathcal{K}(2)$.

and $\frac{3^{g+1}-3}{2}$ edges. The resistance diameter of $\mathcal{H}(g)$ is [50]

$$R(\mathcal{H}(g)) = \frac{4(g-1)}{3} + \frac{2}{3}. \tag{8}$$

Hanoi Graphs. The Hanoi graphs $\mathcal{T}(g)$ are based on the Tower of Hanoi puzzle with g discs [72]. Each legal state of the puzzle corresponds to a node in $\mathcal{T}(g)$, with edges linking nodes whose states can be transformed by moving one disc; $\mathcal{T}(g)$ has 3^g nodes and $\frac{3(3^g-1)}{2}$ edges. Figure 5(b) illustrates the Hanoi graph $\mathcal{T}(3)$. The resistance diameter of $\mathcal{T}(g)$ has been obtained to be [54]

$$R(\mathcal{T}(g)) = \left(\frac{5}{3}\right)^g - 1. \tag{9}$$

Sierpiński Gaskets. The Sierpiński gaskets are constructed iteratively. Starting with an equilateral triangle $\mathcal{S}(0)$, each iteration $g \geq 1$ involves bisecting the edges of all upward-pointing triangles in $\mathcal{S}(g-1)$ and removing the central triangle, resulting in $\mathcal{S}(g)$ with three copies of the previous iteration's triangles. Figure 5(c) illustrates the first three generations of Sierpiński gaskets. There are $\frac{3(3^g+1)}{2}$ nodes and 3^{g+1} edges in graph $\mathcal{S}(g)$. In [31], the resistance diameter of $\mathcal{S}(g)$ is obtained to be

$$R(\mathcal{S}(g)) = \frac{2}{3}\left(\frac{5}{3}\right)^g. \tag{10}$$

Koch Networks. The Koch networks are constructed iteratively [73]. Starting with a triangle $\mathcal{K}(0)$, each iteration $g \geq 1$ involves generating two new nodes for each node in $\mathcal{K}(g-1)$, which are then connected to their "mother" nodes to form new triangles, resulting in $\mathcal{K}(g)$. Figure 5(d) illustrates the construction of the Koch network $\mathcal{K}(2)$. In network $\mathcal{K}(g)$, there are $2 \cdot 4^g + 1$ nodes and $3 \cdot 4^g$ edges. It has been shown [71] that the resistance diameter of the Koch network $\mathcal{K}(g)$ is

$$R(\mathcal{K}(g)) = \frac{2}{3}(2g+1). \tag{11}$$

G OVERVIEW OF BASELINE METHODS

- Optimum: choose k edges from Q by exhaustive search to form the set P that minimize the resistance diameter of $\mathcal{G}(P) = (V, E \cup P)$.
- Random: randomly choose one edge from Q of the updated graph each time. Repeating this process k times, until k edges are added.
- Closeness: choose an edge from Q of the updated graph each time, the two endpoints of which have the lowest values of closeness centrality. Repeating this process k times, until k edges are selected.
- PageRank: choose an edge from Q of the updated graph each time, two endpoints of which have the lowest values of PageRank centrality. Repeating this process k times, until k edges are selected.
- Diameter: choose two farthest nodes of the updated graph and establish a link between them. Repeating this process k times, until k edges are selected.
- Wspd: First obtain k pairs of well-separated point sets in an Euclidean space generated by ApproxER, by using the well-separated pair decomposition in [12]. Let $\{\{A_1, B_1\}, \ldots, \{A_k, B_k\}\}$ be the output of WSPD. Then, choose two nodes u_i and v_i from each set A_i and B_i. The k edges $\{(u_1, v_1), \ldots, (u_k, v_k)\}$ are those to be added to the graph.