**Noname manuscript No.**
(will be inserted by the editor)

# Optimizing Network Robustness by Edge Rewiring: A General Framework

**Hau Chan · Leman Akoglu**

Received: date / Accepted: date

**Abstract** Spectral measures have long been used to quantify the robustness of real-world graphs. For example, spectral radius (or the principal eigenvalue) is related to the effective spreading rate of dynamic processes (e.g., rumor, disease, information propagation) on graphs. Algebraic connectivity (or the Fiedler value), which is a lower bound on the node and edge connectivity of a graph, captures the "partitionability" of a graph into disjoint components.

In this work we address the problem of modifying a given graph's structure under a given budget so as to maximally improve its robustness, as quantified by spectral measures. We focus on modifications based on *degree-preserving* edge rewiring, such that the expected load (e.g., airport flight capacity) or physical/hardware requirement (e.g., count of ISP router traffic switches) of nodes remain unchanged. Different from a vast literature of measure-independent heuristic approaches, we propose an algorithm, called EdgeRewire, which *optimizes* a specific measure of interest directly. Notably, EdgeRewire is *general* to accommodate six different spectral measures. Experiments on real-world datasets from three different domains (Internet AS-level, P2P, and airport flights graphs) show the effectiveness of our approach, where EdgeRewire produces graphs with both (*i*) higher robustness, and (*ii*) higher attack-tolerance over several state-of-the-art methods.

**Keywords** graph robustness · edge rewiring · robustness measures · graph spectrum · optimization algorithms · attack tolerance

## 1 Introduction

Robustness (or resilience, anti-vulnerability, attack-tolerance, connectedness) is a critical property of complex networked systems, such as the Internet, flight networks, and the power grid. It defines their ability to continue functioning in the face of failures or attacks to the parts of the network. As a result, designing and

Hau Chan and Leman Akoglu
Stony Brook University, Department of Computer Science, Stony Brook, NY
Tel.: +1 631-632-9801
E-mail: {hauchan, leman}@cs.stonybrook.edu

maintaining robust networks have been important research areas in physics, engineering, and computer science.

The problem setting we consider in this work is as follows. Given an existing network and a budget, how can we modify the structure of the network in order to maximally improve its robustness while meeting the budget constraints? There are two aspects of this problem that need to be specified: (i) the robustness measure of interest, and (ii) the modification strategies to be employed. The budget is often in terms of the maximum number of modifications that can be performed.

There exists a diverse set of robustness measures studied in the literature [13] (Section 2). Fundamentally, all of these measures try to capture the connectedness of a network in some way. In this work, we specifically focus on a well-established and widely used group of measures, called *spectral* measures, derived from the adjacency and the Laplacian matrices of a network [12]. We build a *general* solution that applies to *six* different spectral measures with small variations in our proposed algorithm (Section 3).

As for modification strategies (Section 4), one can either add new edges [4, 8, 36, 37, 39, 41] or swap (i.e., rewire) existing edges to improve network robustness [4, 25, 32, 35, 40, 43]. For rewiring, one can also either try to preserve the original degrees of the nodes or swap arbitrarily with no such constraint. In this work, we adopt the former, i.e., *degree-preserving* rewiring strategy. The reason is that adding edges, e.g., increasing number of flights between airports would require increased capacity, which might prove impractical in the short term. In fact, numerous examples of infrastructure networks present capacity constraints, such as adding new transmission lines to a power station or new traffic switches to an Internet Service Provider router. Therefore, a rewiring strategy where links are only swapped, keeping the nodes' degrees unchanged, is likely more appropriate; e.g., we reroute Internet traffic among the routers, without considerably changing their load or requiring new hardware with increased number of switches.

A vast majority of network manipulation methods in the literature are measure-independent heuristics. That is, they perform their operations in an intuitive manner (e.g., add edges between two lowest degree nodes), rather than being tied to a particular measure. In other words, they would perform the same operations no matter what the robustness measure of interest is. On the other hand, we propose a solution that aims to directly *optimize* a specified (spectral) measure.

In summary, we propose a new algorithm to modify a given network by degree-preserving edge rewiring so as to maximally improve its robustness under a specified budget (i.e., number of rewirings). We build our solution around a general framework called EdgeRewire that accommodates six different spectrum-based measures, and aims to optimize each measure directly. Experiments on real-world networks from three different domains (Internet AS-level, P2P, and airport flights graphs) show the effectiveness of EdgeRewire, which yields graphs that exhibit both higher robustness as well as higher attack-tolerance compared to graphs produced by several state-of-the-art approaches (Section 5).

## 2 Measures of Network Robustness

For numerous complex networked systems, such as transportation (e.g., road, airline, electric grid), communication (e.g., phone call, email), and computer networks (e.g., the Internet), a critical question is how robust these systems are. Robustness represents the capability of a network to continue its functioning and support its services when parts of the network are naturally damaged or attacked. As such, being able to quantify the robustness of networks becomes important as it allows to measure their vulnerability, compare two networks, modify existing networks to improve their robustness, and design robust networks from scratch.

While centered around intuitively compelling goals, robustness is only vaguely defined: given a graph, quantify how resilient it is to (random or cascading) failures and (targeted and carefully planned) attacks. As a result of not having a unique definition and objective, as well as having been studied in many fields including mathematics, physics, computer science, and biology, various robustness measures have been proposed in the literature. Such measures fundamentally aim to capture the connectedness of a network in some way.

For example, mean shortest paths [2, 4, 20] and efficiency [23, 31] quantify the shortest path distances between pairs of nodes in the network, while Sun *et al.* consider the fraction of connected node pairs [34]. Albert *et al.* use the relative size of the largest connected component and the average size of the other components [2]. Matisziw and Murray consider average available flows between the node pairs [27]. Other measures include the global clustering coefficient, the diameter, node or edge connectivity, etc. [13].

In addition to the above measures that explicitly leverage the graph topology to quantify connectivity, there exist another group of measures derived from the adjacency and the Laplacian matrices of a network [12], called the spectrum-based measures. The graph spectrum has been studied to understand several properties of varying network topologies as well as real-world networks [1, 17, 18]. The spectrum-based measures have been shown to be associated with the inherent interconnectedness, partitionability, and propagation speed and convergence rate of dynamic processes of networks [7, 15, 18], and thus have been widely used to quantify the robustness of networks.

In this work, we focus on this second group of measures and develop a general framework to modify a given network by edge rewiring so as to maximally improve its robustness under a specified budget (i.e., number of rewirings). Our framework is general, as it accommodates a list of different spectrum-based measures. We elaborate on the specific measures after introducing the notation next.

*Notation.* Let $G = (V, E)$ be an undirected graph with $|V| = n$ nodes and $|E| = m$ edges. The topology of $G$ can be described by the adjacency matrix $\mathbf{A}$, an $n \times n$ zero-one matrix, where the element $a_{ij} = 1$ if there is an edge between node $i$ and node $j$, and $a_{ij} = 0$ otherwise. For an undirected graph $G$, $\mathbf{A}$ is symmetric, all its eigenvalues are real, and it exhibits a spectral decomposition $\mathbf{A} = U\Lambda U^T$, where $U = [\mathbf{u_1}\ \mathbf{u_2}\ \ldots\ \mathbf{u_n}]$ is an orthogonal matrix such that $U^TU = UU^T = I$. The $\mathbf{u_k}$'s (columns of $U$) are the eigenvectors of $\mathbf{A}$ with corresponding eigenvalues $\lambda_k$'s (diagonal entries of $\Lambda$). The adjacency spectrum of $G$ is then the set of eigenvalues of $\mathbf{A}$, $\lambda_n \leq \lambda_{n-1} \leq \ldots \leq \lambda_1$, where $\lambda_1$ is the largest eigenvalue.

On the other hand, the Laplacian matrix of $G$ is an $n \times n$ symmetric matrix $\mathbf{L} = D - \mathbf{A}$, where $D = diag(d_i)$ and $d_i$ is the degree of node $i \in V$. As the Laplacian is symmetric, positive semidefinite, and the rows sum to 0, its eigenvalues are real and non-negative, where the smallest one is equal to zero. The Laplacian spectrum of $G$ is then the set of eigenvalues of $\mathbf{L}$, $\mu_1 = 0 \leq \mu_2 \leq \ldots \leq \mu_n$, with corresponding eigenvectors $\mathbf{v_k}$, $k = 1 \ldots n$.

### 2.1 Measures based on the Graph Adjacency Spectrum

In this work we consider three adjacency-spectrum based robustness measures: (1) spectral radius, (2) spectral gap, and (3) natural connectivity.

*1) Spectral radius.* The largest or the principal eigenvalue $\lambda_1$ of the adjacency matrix is called the spectral radius. The effective spreading rate $\tau$ of dynamic processes on a network, such as virus, rumor, or information spread, at which a phase transition occurs, which specifies the onset of the remaining fraction of infected nodes, is found to be proportional to $\tau = \frac{1}{\lambda_1(\mathbf{A})}$ [7]. As a result, spectral radius has been used as a measure of graph vulnerability and robustness, e.g., in [24, 30, 36, 37, 39].

*2) Spectral gap.* The difference between the largest and the second largest eigenvalues of the adjacency matrix, denoted as $\lambda_1 - \lambda_2$, is called the spectral gap. It relates to the expansion properties of the graph [15]. In addition, dynamic processes on graphs converge towards their steady-state, in most cases, exponentially fast in time, with a time-constant that is related to the spectral gap. This measure has been used as a robustness measure, e.g., in [26].

As such, spectral radius and spectral gap, based on the top eigenvalues of the adjacency matrix, are powerful characterizers of dynamic processes on graphs.

*3) Natural connectivity.* This measure is defined [42] as

$$\bar{\lambda}(G) = \ln\left(\frac{1}{n}\sum_{i=1}^{n} e^{\lambda_i}\right) \tag{1}$$

which can be thought of as the "average eigenvalue" of $G$. It is also related to the subgraph centralities ($SC$) in the graph. The $SC(i)$ of a node $i$ is known as its communicability [16], and is based on the "weighted" sum of the number of closed walks that it participates in. The total subgraph centrality of a graph is then $S(G) = \sum_{i=1}^{n} SC(i) = \sum_{i=1}^{n}\sum_{k=0}^{\infty}\frac{(\mathbf{A}^k)_{ii}}{k!}$, where $(A^k)_{ii}$ is the number of closed walks of length $k$ of node $i$. The $k!$ scaling ensures that the weighted sum does not diverge, and longer walks count less.

Noting that $\sum_{i=1}^{n}(\mathbf{A}^k)_{ii} = \text{trace}(\mathbf{A}^k) = \sum_{i=1}^{n}\lambda_i^k$ and by Taylor series of the exponential function we can write

$$S(G) = \sum_{k=0}^{\infty}\sum_{i=1}^{n}\frac{(\mathbf{A}^k)_{ii}}{k!} = \sum_{i=1}^{n}\sum_{k=0}^{\infty}\frac{\lambda_i^k}{k!} = \sum_{i=1}^{n} e^{\lambda_i}\ .$$

As such, natural connectivity can also be thought of as the "average communicability" in $G$. It has been used as a measure of robustness, e.g., in [8, 9].

### 2.2 Measures based on the Graph Laplacian Spectrum

In this work we also consider three Laplacian-spectrum based robustness measures: (1) algebraic connectivity, (2) effective resistance, and (3) spanning tree count.

*1) Algebraic connectivity.* The second smallest, or the first non-zero, eigenvalue $\mu_2$ of the Laplacian is called the algebraic connectivity [18]. As the multiplicity of Laplacian eigenvalues equal to zero is related to the number of connected components, algebraic connectivity is equal to zero for disconnected graphs.

Intuitively, the larger the algebraic connectivity is, the more difficult it is to cut a graph into independent components. Another interpretation for algebraic connectivity comes from node and edge connectivity of the graph $\kappa_v$ and $\kappa_e$, respectively. As it lower-bounds these connectivity metrics, i.e., $0 \leq \mu_2 \leq \kappa_v \leq \kappa_e \leq d_{\min}$, the larger the algebraic connectivity, the larger the number of edges required to disconnect a network and hence, the more robust a network. It has been studied as a robustness measure of a graph, e.g., in [21, 35, 40].

*2) Effective resistance.* This measure is the sum of the effective resistances over all node pairs, which is defined as the electrical resistance seen between the nodes when the graph is treated as a resistor network [19]. It is also equal to the sum of the (inverse) non-zero Laplacian eigenvalues [22]. As such,

$$R = \frac{1}{2}\sum_{i,j}^{n} R_{ij} = n\sum_{i=2}^{n}\frac{1}{\mu_i} \tag{2}$$

Ellens *et al.* [14] propose to use effective resistance as a quantifier for graph robustness; the smaller the effective graph resistance the more robust the network. The authors make several arguments as to why it qualifies as a robustness measure. First, similar to natural connectivity in Eq. (1), it takes both the number of paths between node pairs and their length into account, therefore the number of backup paths as well as their quality is considered. Second, effective graph resistance can be approximated by the algebraic connectivity, which is used as a measure for network robustness. Moreover, it relates to the distance between the nodes. As Chandra *et al.* [10] shows, effective resistance can be written in terms of the expected commute time between node pairs;

$$R_{ij} = \frac{1}{2m}(\mathbb{E}(T_{ij}) + \mathbb{E}(T_{ji}))$$

where $T_{ij}$ denotes the number of transitions (i.e., hitting time) to reach node $j$ starting in $i$.

*3) Number of spanning trees.* Baras and Hovareshti [3] suggest the number of spanning trees[1] as an indicator of network robustness. As a consequence of the Kirchhoff's matrix-tree theorem, the number of non-identical spanning trees can be written as a function of the unweighted Laplacian eigenvalues [6] as

$$S = \frac{1}{n}\prod_{i=2}^{n}\mu_i \tag{3}$$

## 2.3 Discussion on Spectrum-based Measures

The spectrum-based measures we consider in this work are well-established and widely-used measures [3, 8, 9, 14, 21, 24, 26, 30, 35, 36, 37, 39, 40]. They are good indicators for the robustness of a given graph, as they relate closely to properties such as connectivity (a.k.a. path capacity), expansion, propagation dynamics, etc.

[1] A spanning tree is a subgraph over all nodes, containing ($n$-1) edges and no cycles.

The choice of a specific measure is primarily application-dependent. For example, trying to reduce propagation of a disease on a network would aim to modify a graph's structure to maximally decrease its spectral radius. To reduce the number of alternative paths in a communication system (e.g., among terrorists), one may adopt natural connectivity or effective resistance.

While the measure to be used depends on the application, there exist intuitive criteria one could think of that a reasonable robustness measure should satisfy. For example, it would be desired that the measure increases when new edges are added to a given network. In other words, adding more infrastructure that potentially costs resources (e.g., \$) should ideally (strictly) improve the robustness of the network. It would also be desirable for the measure to be able to quantify the robustness of *different* graphs with multiple connected components.

In retrospect, we can show that effective resistance and natural connectivity are strictly monotonic [14], while algebraic connectivity is only monotonic (monotonicity follows directly from the interlacing theorem [11]). On the other hand, effective resistance cannot be computed for disconnected graphs, for which it is defined to be infinity, whereas algebraic connectivity is zero for all disconnected graphs and hence cannot differentiate the robustness of different graphs with multiple components. Natural connectivity, on the other hand, could yield different values for different disconnected graphs, since the spectrum of a disconnected graph is the union of the spectra of all of its individual components [5].

The list of desired properties, some examples of which are discussed above, could be compiled under a set of *axioms* (e.g., strict monotonicity) and a formal analysis of which measures satisfy which desired properties (i.e., axioms) can be carried out. Those could then serve as guidelines for applications into the choice of a measure. As such, an axiomatic analysis of spectral measures and others is a very promising research direction. However, providing a formal analysis that could guide toward the choice of a specific measure is outside the scope of this work. Our goal is to develop a framework for manipulating the graph structure by (degree-preserved) edge-rewiring—*provided* that a spectral measure is already chosen to quantify the graph robustness.

## 3 Optimizing Spectrum-based Network Robustness by Edge Rewiring

Given a graph and a robustness measure of interest, how can we rewire a few of the edges such that its robustness is maximally increased? In this work, we consider this general question under two specific criteria: (1) we focus on the *spectrum based* measures, and (2) we aim for solutions that *preserve* the node degrees.

The reason we focus on spectrum based measures is that they are extremely effective in capturing the inherent connectedness of a network, as described in detail in the previous section. They relate to phase transitions, spreading speed, and convergence rates of dynamical processes on graphs, capture node centralities, resistance and distances between node pairs, and partitionability at large. As such, they have wide applicability as robustness measures in practice.

The rationale for optimizing robustness via degree-preserving edge rewiring is related to management costs. To improve robustness, one can consider adding new edges, which would change the degree (i.e., the load) of several nodes in the network. Most infrastructure networks, however, present a capacity constraint.

For example, adding new transmission lines to a power station, new flight routes to an airport, or new traffic cables to an Internet Service Provider are all expected to incur additional management and hardware costs compared to rerouting power, flights, or traffic by swapping a few links, where degrees (i.e., expected load) are preserved. As such, optimizing robustness based on (degree-preserving) edge rewiring is often more appropriate and less costly than edge addition in practice.

Overall, the problem statement we consider can be written as follows.

**Edge Rewiring Problem:**

**Given** a graph $G$, a robustness measure $r$, and an integer budget $k$;
**Find** $k$ pairs of edges, the rewirings of which maximally increase $r(G)$.

The rewiring of a pair of edges $(i, j) \in E$ and $(p, r) \in E$ involves removing these edges and adding either edges $(i, p) \notin E$ and $(j, r) \notin E$ or edges $(i, r) \notin E$ and $(j, p) \notin E$, whichever yields a higher $r$. This scheme ensures that the degrees of the nodes are preserved.

The spectrum based robustness measures use either some or all of the eigenvalues of the adjacency or the Laplacian matrix of the graph. When manipulations are introduced to the graph by edge rewiring, the spectrum of the graph changes. As such, the measures need to be updated accordingly. Computing the eigenvalues from scratch for each possible manipulation to be done on the graph, however, is very costly. Ideally, one would only update the eigenvalues incrementally. In the following we show that one can compute the changes to eigenvalues (and eigenvectors) efficiently using the first order matrix perturbation theory [33].

## 3.1 Updating the Graph Spectrum by Matrix Perturbation Theory

Let $(\alpha_j, \mathbf{x_j})$ be the $j^{th}$ (eigenvalue, eigenvector) pair of a symmetric $(n \times n)$ matrix $\mathbf{M}$. Let $\Delta\mathbf{M}$ and $(\Delta\alpha_j, \Delta\mathbf{x_j})$ denote the change in $\mathbf{M}$ and $(\alpha_j, \mathbf{x_j})$ $\forall j$, respectively, when entries of $\mathbf{M}$ are modified. Suppose after the modifications $\mathbf{M}$ becomes

$$\tilde{\mathbf{M}} = \mathbf{M} + \Delta\mathbf{M}$$

where $(\tilde{\alpha}_j, \tilde{x}_j)$ is written as

$$\tilde{\alpha}_j = \alpha_j + \Delta\alpha_j \qquad \text{and} \qquad \tilde{\mathbf{x}}_j = \mathbf{x_j} + \Delta\mathbf{x_j}$$

**Lemma 1** *Given a perturbation $\Delta\mathbf{M}$ to a matrix $\mathbf{M}$, its eigenvalues can be updated by*

$$\Delta\alpha_j = \mathbf{x_j}^T \Delta\mathbf{M}\, \mathbf{x_j}. \tag{4}$$

**Lemma 2** *Given a perturbation $\Delta\mathbf{M}$ to a matrix $\mathbf{M}$, its eigenvectors can be updated by*

$$\Delta\mathbf{x_j} = \sum_{i=1, i\neq j}^{n} \left( \frac{\mathbf{x_i}^T \Delta\mathbf{M}\, \mathbf{x_j}}{\alpha_j - \alpha_i} \mathbf{x_i} \right). \tag{5}$$

*Proof* Proofs of Lemma 1 and Lemma 2 are omitted for brevity. See [33].

## 3.2 Updating the Spectrum-based Measures under Edge Manipulation

Rewiring the edges of a given graph involves deleting existing edges and introducing new edges. In order to choose promising edges that will have high positive impact on a specified robustness measure $r$, we need to quantify the effect an edge addition/deletion would have on robustness, i.e. the increase/decrease in $r$. In the following, we discuss how robustness changes for edge addition for the six measures in §2.1 and §2.2. Similar but negated equations follow for edge deletion.

*3.2.1 Spectral radius.* When an edge $(i, j)$ is added to a graph, the change in its spectral radius $\Delta\lambda_1$ can be written as

$$\Delta\lambda_1 = \mathbf{u_1}^T \Delta\mathbf{A}\ \mathbf{u_1} = 2\mathbf{u_{1i}}\mathbf{u_{1j}} \tag{6}$$

based on Eq. (4) of Lemma 1, where $\mathbf{M}$ is replaced with the adjacency matrix $\mathbf{A}$. That is, $\Delta\mathbf{A}(i, j) = \Delta\mathbf{A}(j, i) = 1$ and 0 elsewhere.

*3.2.2 Spectral gap.* The change in spectral gap when an edge $(i, j)$ is added can be denoted as $\Delta\lambda_1 - \Delta\lambda_2$ and can be written similarly as

$$\Delta\lambda_1 - \Delta\lambda_2 = \mathbf{u_1}^T \Delta\mathbf{A}\ \mathbf{u_1} - \mathbf{u_2}^T \Delta\mathbf{A}\ \mathbf{u_2} = 2(\mathbf{u_{1i}}\mathbf{u_{1j}} - \mathbf{u_{2i}}\mathbf{u_{2j}}) \tag{7}$$

*3.2.3 Natural connectivity.* When an edge $(i, j)$ is added, we can write the change in natural connectivity $N$ as

$$\begin{aligned}
\Delta N &= \ln\,(\frac{1}{n}\sum_{j=1}^{n} e^{\lambda_j + \Delta\lambda_j}) - N \\
&= \ln\,(e^{\lambda_1}(e^{\Delta\lambda_1} + e^{(\lambda_2-\lambda_1)}e^{\Delta\lambda_2} + \ldots + e^{(\lambda_n-\lambda_1)}e^{\Delta\lambda_n})) - \ln\ n - N \\
&= \ln\,(e^{2\mathbf{u_{1i}}\mathbf{u_{1j}}} + e^{(\lambda_2-\lambda_1)}e^{2\mathbf{u_{2i}}\mathbf{u_{2j}}} + \ldots + e^{(\lambda_n-\lambda_1)}e^{2\mathbf{u_{ni}}\mathbf{u_{nj}}}) + \ln\ \frac{e^{\lambda_1}}{n} - N \\
&\approx \ln\,(e^{2\mathbf{u_{1i}}\mathbf{u_{1j}}} + e^{(\lambda_2-\lambda_1)}e^{2\mathbf{u_{2i}}\mathbf{u_{2j}}} + \ldots + e^{(\lambda_t-\lambda_1)}e^{2\mathbf{u_{ti}}\mathbf{u_{tj}}}) + \ln\ \frac{e^{\lambda_1}}{n} - N \quad (8)
\end{aligned}$$

Notice that the coefficients $e^{(\lambda_k-\lambda_1)}$ are getting smaller with increasing $k$ as the gaps between the largest and the successive eigenvalues get larger. In fact, real-world graphs have been observed to exhibit a very skewed, power-law-like spectrum where only the top few eigenvalues have large magnitude, which drops super-linearly fast for increasing $k$ [17]. Moreover, if the graph is robust it is expected that $(\lambda_2 - \lambda_1)$, i.e., the spectral gap, is already large. This makes it possible to approximate the natural connectivity with the top $t$ eigenvalues with the highest algebraic magnitude for a considerably small $t$. A rule of thumb is to set $t = 30$ as suggested in [38] where the skewed spectrum of real graphs has been leveraged to approximate local triangle counts. Finally, for edge deletions we replace the $2\mathbf{u_{ti}}\mathbf{u_{tj}}$ terms with $-2\mathbf{u_{ti}}\mathbf{u_{tj}}$, for $k = 2 \ldots t$.

*3.2.4 Algebraic connectivity.* When an edge $(i, j)$ is added to a graph, the change in its algebraic connectivity $\Delta\mu_2$ can be written as

$$\Delta\mu_2 = \mathbf{v_2}^T \Delta\mathbf{L}\ \mathbf{v_2} = (\mathbf{v_{2i}} - \mathbf{v_{2j}})^2 \tag{9}$$

based on Eq. (4) of Lemma 1, where $\mathbf{M}$ is replaced with the Laplacian matrix $\mathbf{L}$, $\Delta\mathbf{L}(i, j) = \Delta\mathbf{L}(j, i) = -1$, $\Delta\mathbf{L}(i, i) = \Delta\mathbf{L}(j, j) = 1$, and 0 elsewhere.

*3.2.5 Effective resistance.* When an edge $(i, j)$ is added, we can write the change in effective resistance $R$ as

$$\begin{aligned}
\Delta R &= n\ (\frac{1}{\mu_2 + \Delta\mu_2} + \frac{1}{\mu_3 + \Delta\mu_3} + \ldots + \frac{1}{\mu_n + \Delta\mu_n}) - R \\
&= n\ (\frac{1}{\mu_2 + (\mathbf{v_{2i}} - \mathbf{v_{2j}})^2} + \frac{1}{\mu_3 + (\mathbf{v_{3i}} - \mathbf{v_{3j}})^2} + \ldots + \frac{1}{\mu_n + (\mathbf{v_{ni}} - \mathbf{v_{nj}})^2}) - R \\
&\approx n\ (\frac{1}{\mu_2 + (\mathbf{v_{2i}} - \mathbf{v_{2j}})^2} + \ldots + \frac{1}{\mu_t + (\mathbf{v_{ti}} - \mathbf{v_{nj}})^2}) - R
\end{aligned} \tag{10}$$

Similar to natural connectivity, effective resistance requires all the eigenvalues in the spectrum. However, computing them all would be very expensive, especially for large graphs with large $n$. As the eigenvalues of the graph Laplacian are all non-negative, notice that the denominators $(\mu_k + \Delta\mu_k)$ are expected to get larger with increasing $k$, and hence yield smaller terms. As such, one can approximate $\Delta R$ with the bottom $t$ eigenvalues with the smallest magnitude. For edge deletions, we replace the denominators with $(\mu_k - (\mathbf{v_{ki}} - \mathbf{v_{kj}})^2)$ for $k = 2 \ldots t$.

*3.2.6 Number of spanning trees.* When an edge $(i, j)$ is added, we can write the change in the number of spanning trees $S$ as

$$\begin{aligned}
\Delta S &= \frac{1}{n}\ (\mu_2 + \Delta\mu_2)\ (\mu_3 + \Delta\mu_3)\ \ldots\ (\mu_n + \Delta\mu_n) - S \\
&= \frac{1}{n}\ (\prod_{i=2}^{n} \mu_i)\ \frac{(\mu_2 + \Delta\mu_2)}{\mu_2}\ \frac{(\mu_3 + \Delta\mu_3)}{\mu_3}\ \ldots\ \frac{(\mu_n + \Delta\mu_n)}{\mu_n} - S \\
&\approx S\ (\frac{\mu_2 + (\mathbf{v_{2i}} - \mathbf{v_{2j}})^2}{\mu_2}\ \frac{\mu_3 + (\mathbf{v_{3i}} - \mathbf{v_{3j}})^2}{\mu_3}\ \ldots\ \frac{\mu_t + (\mathbf{v_{ti}} - \mathbf{v_{tj}})^2}{\mu_t} - 1)
\end{aligned} \tag{11}$$

Note that this measure also requires the entire spectrum, which is expensive to compute for large $n$. Notice that increasing the smaller eigenvalues would yield larger terms in the multiplication and hence yield higher increase in $S$, while the terms involving larger eigenvalues would be close to one. Therefore, one can focus on the smallest $t$ eigenvalues of the Laplacian for efficiency. Finally, we replace the numerators with $(\mu_k - \Delta\mu_k)$ for edge deletions for $k = 2 \ldots t$.

Notice that for the last two measures, we suggest to use the $t$ smallest eigenvalues of the Laplacian for efficient computation. The justification of using the $t$ smallest Laplacian eigenvalues is because real-world graphs have been observed to exhibit a very skewed, power-law-like Laplacian spectrum where only the bottom few eigenvalues have small magnitude (see Figure 1).

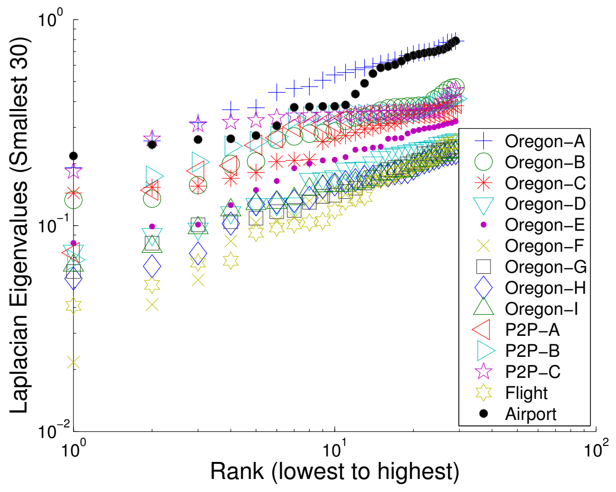

**Fig. 1** Bottom $t = 30$ smallest eigenvalues versus rank (log-log) of the Laplacian matrix of real-world graphs (See § 5.2 for dataset descriptions).

## 3.3 Proposed Edge Rewiring Algorithm

In this work, we strive to answer the question of which $k$ edge pairs should be rewired to increase a given spectral measure of interest the most, while preserving node degrees. Given a graph $G = (V, E)$ with $|E| = m$ edges, the number of possibilities to rewire two edges is given by $2\binom{m}{2}$. This is quadratic in the number of edges, and hence is infeasible for large graphs, even for $k = 1$.

For this reason, we propose a strategy that performs an edge rewiring as a series of edge additions and edge deletions. In the previous section we showed how the six robustness measures that we consider change if a single edge were to be added (or deleted). As such, for the edge addition (deletion) problem, one would simply choose the edge with the largest (smallest) $\Delta r$. For the general problem of selecting the optimal *set of* $k$ edges to add (delete) to maximally increase (decrease) the robustness of an input graph, however, is often NP-hard. In fact, it has been proven that adding/deleting a specified number of edges to a graph to maximize/minimize its algebraic connectivity is NP-hard [28], and so it is for its spectral radius [39].

As the set selection problem poses combinatorial challenges, and given that the complimentary problem of optimal edge addition/deletion is NP-hard (at least for several of our measures), we propose a heuristic iterative algorithm, called EdgeRewire. Our approach is based on searching for the edge pairs one by one and performs each rewiring incrementally step by step in a best-first search fashion. Specifically, we rewire one pair of edges in a three step process and repeat this process $k$ times; (1) add non-existing edge $(i, p)$ that increases robustness the most, (2) remove edge $(i, j)$ (or $(p, r)$) that decreases robustness the least, and (3) add non-existing edge $(j, r)$ and remove edge $(p, r)$ (or remove $(i, j)$) that increases robustness the most (or decreases it the least). These steps are illustrated in Figure 2, and the general outline of EdgeRewire is given in Algorithm 1 which we describe in more detail as follows.

**Algorithm 1** EdgeRewire

```
Input: Graph $G = (V, E)$, robustness measure $r$, integer budget $k$
Output: Updated graph $\tilde{G}$ with $k$ pairs of edges rewired (degrees preserved)
 1: if $r$ is based on adjacency spectrum then
 2:     Compute top $t$ (eigenvalue, eigenvector) pairs $(\lambda_k, \mathbf{u_k})$ of $\mathbf{A}$, $1 \leq k \leq t$
 3: else if $r$ is based on Laplacian spectrum then
 4:     Compute bottom $t$ (eigenvalue, eigenvector) pairs $(\mu_k, \mathbf{v_k})$ of $\mathbf{L}$, $2 \leq k \leq t$
 5: end if
 6: for $iter = 1$ to $k$ do
 7:     Select edge $(i, p)$ to add, s.t. $(i, p) \notin E$, $i \in V$, $p \in V$, $i \neq p$, that maximizes $\Delta r_1$:
            Eq. (6) {if $r$ is spectral radius $\lambda_1$}
            Eq. (7) {if $r$ is spectral gap $\lambda_1 - \lambda_2$}
            Eq. (8) {if $r$ is natural connectivity $N$}
            Eq. (9) {if $r$ is algebraic connectivity $\mu_2$}
            Eq. (10) {if $r$ is effective resistance $R$}
            Eq. (11) {if $r$ is number of spanning trees $S$}
 8:     Select edge $(i, j)$ or edge $(p, r)$ to remove, such that $j \in \mathcal{N}(i)$ and $r \in \mathcal{N}(p)$, that
        maximizes $\Delta r_2$ (note that $\Delta r_2$ is negative) (suppose edge $(i, j)$ it is)
 9:     Select node $r \in \mathcal{N}(p)$ for which removing edge $(p, r)$ and adding edge $(j, r)$ maximizes
        total $\Delta r_3$ (note that $\Delta r_3$ may be positive or negative)
10:     Repeat Lines 7-9 with next edge $(i, p)$ with highest $\Delta r_1$, until $\Delta r_1 + \Delta r_2 + \Delta r_3 > 0$
11:     $\mathbf{A}(i,p)$=$\mathbf{A}(p,i)$=1, $\mathbf{A}(j,r)$=$\mathbf{A}(r,j)$=1, $\mathbf{A}(i,j)$=$\mathbf{A}(j,i)$=0, $\mathbf{A}(p,r)$=$\mathbf{A}(r,p)$=0
12:     Update $t$ eigenvalues by Eq. (4)
13:     Update $t$ eigenvectors by Eq. (5)
14: end for
```

We emphasize that our proposed algorithm is general and accommodates a list of six spectrum-based measures. Depending on the matrix ($\mathbf{A}$ or $\mathbf{L}$) a given measure $r$ is based on, we start by computing the related eigen-pairs—top $t$ for the adjacency based measures in §2.1 (Line 2) and bottom $t$ for the Laplacian based ones in §2.2 (Line 4). Note that we compute the $t$ eigen-pairs for all measures, including spectral radius $\lambda_1$, spectral gap $(\lambda_1 - \lambda_2)$, and effective resistance $\mu_2$, even though they do not use the whole spectrum for reasons we will discuss later in this section.

*Main rewiring steps:* We proceed by selecting the edge pairs to be rewired one by one in an iterative way (Line 6). Within each iteration we first find the best non-existing edge to add that would increase robustness $r$ the most (Line 7). We denote this edge by $(i, p)$. As the edge addition increases the degrees of $i$ and $p$ by 1, we search in the local neighborhood $\mathcal{N}(i)$ of $i$ and $\mathcal{N}(p)$ of $p$ for an edge, the removal of which would reduce $r$ the least (Line 8) (note that our spectral measures are all monotonic). Suppose the selected edge is $(i, j)$. Removal of $(i, j)$ brings degree of $i$ to its original value, while decreases $j$'s by 1. Next, we find the neighbor $r$ of $p$ where the removal of edge $(p, r)$ *and* addition of edge $(j, r)$ increases $r$ the most or decreases $r$ the least (i.e., yields largest change) (Line 9). Figure 2 illustrates these steps. If the total $\Delta r$ from steps 7-9 is positive, we commit all the changes (note that all degrees would be preserved). Otherwise we discard all modifications, and continue with the next best edge from Line 7.

*Updating the eigen-pairs:* For all successful iterations, the graph structure changes (Line 11) and hence the $t$ eigen-pairs as well. Instead of recomputing the eigen-pairs every time, we update them using Lemma 1 and 2 for efficiency (Lines 13-14). Notice that computing the update $\Delta r$ for each of our measures involves the

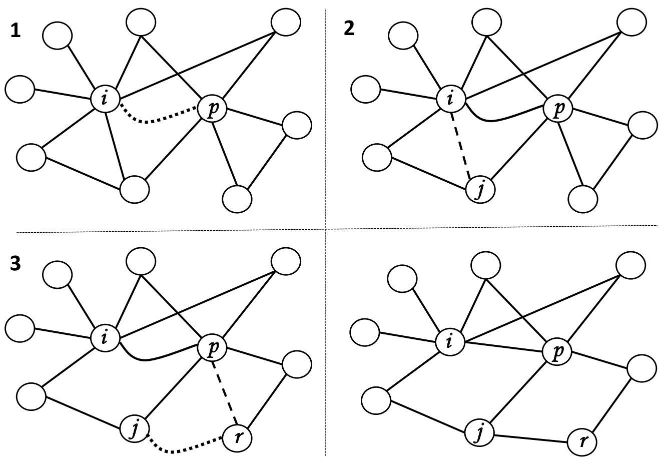

**Fig. 2** Illustration of the main rewiring steps (Lines 7, 8, and 9) of Algorithm 1 on a toy graph. Also shown is the resulting graph after $k = 1$ degree-preserving rewiring.

corresponding eigenvector(s) (Eq. (6) through Eq. (11)). That is the reason why we need to also update the eigenvectors when the graph structure changes. According to Eq. (5) of Lemma 2, updating an eigenvector involves all the other eigen-pairs. For efficiency, one can use only the top/bottom $t$ depending on the measure. Either way, we need to compute (and later update) $t$ eigen-pairs at every iteration, even if the measure of interest involves less eigenvalues like $\lambda_1$, $(\lambda_1 - \lambda_2)$ and $\mu_2$, hence Lines 2 and 4.

*Selecting the first non-existing edge:* The first main step of EdgeRewire is to find a disconnected pair of nodes $i$ and $p$, such that the addition of edge $(i,p)$ increases the robustness more than any other non-existing edge of the graph. For each such edge, the increase it would incur on the robustness when added to the graph is computed, depending on the measure, with respective formulas as listed in Line 7 of Algorithm 1. The increase $\Delta r$ an edge incurs can be thought as its utility score. We compute $\Delta r$ for all non-existing edges of the graph, and sort them in decreasing order of this score. We perform this step of the algorithm (i.e., select a non-existing edge to add), for each rewiring, by selecting the edge at the top of this ordered list, i.e., the edge that *maximizes* robustness. We skip an edge and continue with the next one in the list, when the rewiring does not increase robustness overall (as in Line 10).

*Speeding up the selection:* For sparse real world graphs, the number of non-existing edges is often significantly larger than the number of existing edges. As a result, computing $\Delta r$ for all non-existing edges would be computationally demanding for large graphs. To speed up the selection, one can build an approximate but promising ordering by considering only a subset of the non-existing edges.

For example, consider the spectral radius measure, where $\Delta r(i,p) = 2\mathbf{u_{1i}}\mathbf{u_{1p}}$. Since we are interested in edges with large $\Delta r$, we can first sort the entries, i.e. nodes, of $\mathbf{u_1}$ in decreasing order, select the top $l$ nodes, and consider all pairings of these nodes. This would yield approximately $l^2$ candidate edges with largest $\Delta r$, excluding already-existing edges. The value $l$ can be chosen depending on the desired candidate edge set size, however, often a small value would be enough as the number of candidate edges produced is quadratic in $l$. As another example, consider the algebraic connectivity, where $\Delta r(i,p) = (\mathbf{v_{2i}} - \mathbf{v_{2p}})^2$. In this case, one can sort the nodes by $\mathbf{v_2}$ and build a set of candidate edges by pairing the top few nodes having the largest values with the bottom few nodes having the smallest values. Similar approximations can be done for other measures in order to build a significantly smaller, nevertheless promising set of candidate non-existing edges.

In our implementation, we reconstruct a ranked list of candidate edges for every rewiring in Line 7—as eigenvectors are updated, the ordering of nodes could change. To speed up selection further, one can perform this step only once globally, i.e., outside the loop before Line 6. Assuming the rewirings do not change the eigenvectors much, and especially the ordering of nodes, a global ranking would be quite close to the individual rankings over the iterations.

Note that the computational challenge of choosing the best edge arises only in this first step—when we aim to find the best non-existing edge to add to the graph. For the other steps, the search to add/delete edges is performed locally, in the neighborhood of the nodes that we connect, which is considerably faster.

*Remarks:* Before we conclude, we comment on several aspects of EdgeRewire. In the proposed algorithm, we start by adding the best non-existing edge and build the following steps around it. An alternative way of rewiring is to start by removing the existing edge that would decrease robustness the least and build the solution around it. Experiments with this approach, however, produced notably inferior results. This is potentially due to the fact that such an edge initially picked to be removed lives in the periphery of the graph, connecting low-degree nodes, and thus building a solution at the edge of the network gives poor results. Another remark is about the incremental construction of each rewiring, through three steps (Lines 7, 8, and 9). As the decision at each step is local, the end result may not always yield a positive total difference in robustness, in which case steps 7-9 are repeated until a plausible solution is found.

## 4 Related Work and Alternative Approaches

We covered related work on measures of graph robustness in §2. Therefore, in this section, we discuss related work on graph manipulation algorithms for optimizing robustness. The main type of manipulations include (1) the addition/deletion of edges and/or nodes, and (2) the rewiring of edges.

### 4.1 Edge/Node Addition/Deletion

These manipulation algorithms can be organized in various ways. For instance, one group focuses on improving graph robustness by edge additions, whereas another aims to degrade robustness by carefully removing edges and/or nodes. The algorithms can also be grouped based on their approaches. While most of them are intuitive heuristic based approaches, several others involve optimization schemes.

**Table 1** Comparison of related work. MIH: measure-independent heuristic, Opt: optimization-based approach. Measures are those that the methods in the Opt-category aim to optimize.

| Related work | Node/edge deletion | Edge addition | Edge rewiring | Degree-preserving | MIH / Opt | Measures |
|---|---|---|---|---|---|---|
| [2], [20] | ✓ | | | | MIH | |
| [4] | | ✓ | ✓ | | MIH | |
| [37], [39] | ✓ | | | | Opt | spectral radius $\lambda_1$ |
| [36] | ✓ | ✓ | | | Opt | $\lambda_1$ |
| [41] | | ✓ | | | Opt | algebraic connectivity $\mu_2$ |
| [19] | ✓ | | | | Opt | effective resistance $R$ |
| [8] | ✓ | ✓ | | | Opt | natural connectivity $N$ |
| [25], [32], [43] | | | ✓ | ✓ | MIH | |
| [35] | | | ✓ | | Opt | $\mu_2$ |
| [40] | | | ✓ | ✓ | Opt | $\lambda_1$ and $\mu_2$ |
| EDGEREWIRE | | | ✓ | ✓ | Opt | 6 measures (§2.1 & §2.2) |

The most frequently studied manipulation heuristic to degrade robustness has been the removal of most connected (i.e., highest degree) nodes [2, 4, 20]. Holme *et al.* [20] includes a comparative study where they show that removing nodes/edges based on betweenness centrality is better than degree centrality, especially for removal of edges. Beygelzimer *et al.* [4] investigate modification schemes to improve network robustness. They study edge addition strategies based on (1) random and (2) preferential schemes, and find that preferential edge addition, specifically connecting the lowest degree nodes, yields the best overall results.

The above strategies are heuristics that modify a given graph structure irrespective of a robustness measure. That is, they select nodes/edges based on intuitive criteria (e.g., centrality) rather than directly optimizing a particular measure. As such, they would select the same set of nodes/edges to manipulate for any robustness measure of interest. Unlike these measure-independent heuristics, several approaches involve optimization techniques to systematically manipulate a specific robustness measure, such as spectral radius [36, 37, 39], algebraic connectivity [41], effective graph resistance [19], and natural connectivity [8].

The algorithms proposed in all of these works focus on addition/deletion of nodes/edges or altering the edge weights.

## 4.2 Edge Rewiring

Similar to the above, edge rewiring approaches can be grouped into heuristic versus optimization based solutions. Moreover, several aim to perform degree-preserving rewiring whereas others do not consider such a constraint.

Schneider *et al.* [32] introduce a new robustness measure in terms of the relative size of the largest component during iterative malicious attacks, and propose a simple degree-preserving rewiring scheme to improve this measure. In particular, two randomly selected edges are swapped if the robustness with respect to the node attacks is increased. Zeng *et al.* [43] adopt their measure and use the same rewiring scheme to improve robustness with respect to edge based attacks.

Surprisingly, the robustness optimization by Schneider *et al.* [32] reveal an emerging "onion-like" topology that the resulting networks exhibit, consisting of a core of highly connected nodes hierarchically surrounded by rings of nodes with decreasing degree. Such a structure resembles highly assortative networks [29], in which nodes of similar degrees are connected. Louzada *et al.* [25] use this insight to develop a new rewiring approach, which they call smart rewiring, that is likely to increase assortativity by connecting two average degree nodes. Their rewiring approach also decreases the emphasis on hubs such that their removal would not have huge impact on the global connectivity. Their scheme is also degree-preserving.

Beygelzimer *et al.* [4] also study a list of rewiring heuristics. Preferential rewiring disconnects a random edge from the highest degree node, preferential random edge rewiring disconnects a random edge from its higher degree node, and random rewiring removes a random edge—after which all add an edge by connecting two random nodes. They find preferential schemes to perform better. We note that none of their schemes is degree-preserving.

All of the rewiring strategies to this end are measure-independent heuristics. As for optimization-based approaches, Sydney *et al.* propose a rewiring algorithm for optimizing algebraic connectivity, however their approach is not degree-preserving [35]. Closest to our work are the degree-preserving rewiring approaches proposed by Van Mieghem *et al.* for optimizing spectral radius $\lambda_1$ and algebraic connectivity $\mu_2$ [40]. They analyze Newman's assortativity $\rho$ [29] and provide theoretical bounds that show $\rho$ to correlate positively with $\lambda_1$ and negatively with $\mu_2$. Based on their analysis they propose an assortative rewiring algorithm to improve $\lambda_1$ which however decreases $\mu_2$, and a disassortative one with the opposite effects.

We compare EdgeRewire to the approaches proposed by [25], [32], and [40] in the experiments, where we describe them in more detail. Table 1 provides a summary and comparison of related work.

## 5 Evaluation

The goals of this section are to show (1) that our EdgeRewire algorithm is significantly more effective in improving the robustness of a given graph as compared to several state-of-the-art approaches, and (2) that our rewired graphs are less vulnerable to attacks than graphs rewired through these other methods.

EdgeRewire is applicable to all six spectrum based measures introduced in §2. Two of the measures, particularly $\lambda_1$ and $\mu_2$, have been studied extensively as graph robustness measures, for which there exist several state-of-the-art degree-preserving rewiring approaches proposed in the literature that we use as competitors or baselines. For the other four spectrum based measures (i.e., spectral gap, natural connectivity, effective resistance, and number of spanning trees), we are the first to consider them in the context of degree-preserving edge-rewiring. Nevertheless, we apply the methods earlier proposed for $\lambda_1$ and $\mu_2$ for manipulating those measures as well and compare EdgeRewire to their results. Note that these baselines are not expected to perform well for these four measures as they were not designed for them in the first place, however since we are the first to consider these latter four measures, we use them as baselines for the sake of comparison. Next we briefly describe these existing approaches, and then compare their performances against EdgeRewire on real-world graphs.

**Table 2** Dataset summary; 3 categories of datasets used in evaluation (AS router, P2P, and flight networks) where rewiring is applicable. $|V|$: number of nodes, $|E|$: number of edges. Datasets listed in increasing size order within each category. $\lambda$: spectral radius, $\mu$: algebraic connectivity. All publicly available, see §5.2 for links to the datasets.

| **Dataset** | $\|V\|$ | $\|E\|$ | $\lambda$ | $\mu$ |
|---|---|---|---|---|
| *Oregon-A* (O-A) | 633 | 1,086 | 20.61 | 0.19 |
| *Oregon-B* (O-B) | 1,503 | 2,810 | 28.74 | 0.13 |
| *Oregon-C* (O-C) | 2,504 | 4,723 | 35.19 | 0.14 |
| *Oregon-D* (O-D) | 2,854 | 4,932 | 35.98 | 0.08 |
| *Oregon-E* (O-E) | 3,995 | 7,710 | 40.65 | 0.08 |
| *Oregon-F* (O-F) | 5,296 | 10,097 | 43.06 | 0.02 |
| *Oregon-G* (O-G) | 7,352 | 15,665 | 51.19 | 0.06 |
| *Oregon-H* (O-H) | 10,860 | 23,409 | 56.84 | 0.06 |
| *Oregon-I* (O-I) | 13,947 | 30,584 | 61.65 | 0.06 |
| *P2P-A* (P-A) | 6,301 | 20,777 | 28.38 | 0.07 |
| *P2P-B* (P-B) | 8,114 | 26,013 | 28.45 | 0.07 |
| *P2P-C* (P-C) | 8,717 | 31,525 | 22.38 | 0.18 |
| *Flight* | 2,939 | 15,677 | 63.00 | 0.04 |
| *Airport* | 1,574 | 34,430 | 112.35 | 0.22 |

## 5.1 Existing edge rewiring approaches as baselines

To show the effectiveness of EdgeRewire, we compare the robustness values of our rewired graphs to those of graphs rewired by the following existing heuristics.

1. Assortative (Van Mieghem *et al.* [40]): Randomly select two edges with four different nodes. Rewire the edges such that the two highest-degree nodes and the two lowest-degree nodes are connected.
2. Disassortative (Van Mieghem *et al.* [40]): Randomly select two edges with four different nodes. Rewire the edges such that the highest degree node is linked to the lowest degree node, and the remaining nodes are connected to each other.
3. Smart (Louzada *et al.* [25]): Randomly select a node $i$ with at least two neighbors where their degrees are larger than one. Select the lowest degree neighbor, say $j$, and the highest degree neighbor, say $k$, of $i$. Randomly select a neighbor of $j$, say $m$, and a neighbor of $k$, say $n$. Rewire the edges such that $j$ and $k$ and $m$ and $n$ are connected while $j$ and $m$ and $k$ and $n$ are disconnected.
4. Random (Schneider *et al.* [32]): Randomly select two edges and swap two ends arbitrarily in one of two ways that preserve the degrees.

We note that the above approaches do not explicitly check whether each rewiring achieves an increase in robustness. As such, we also consider variants of them that commit a rewiring only if it increases the measure, or else repeat the random selection. In other words, we require each heuristic to check for improvement of the robustness measure under consideration at every step. Overall, we compare to eight different baseline heuristics.

## 5.2 Dataset description

We use three different categories of datasets to evaluate the methods as shown in Table 2. Those include AS router, P2P, and flight networks for which edge rewiring is meaningful and applicable (e.g., rewiring flights is seamless compared to rewiring road networks). All of our datasets are publicly available at `http://snap.stanford.edu/data/` (Oregon and Gnutella) and at `http://konect.uni-koblenz.de/networks/` (OpenFlights and US Airports).

- **Oregon Autonomous System (AS) Router Networks** These networks are AS-level connectivity networks inferred from Oregon route-views, and were collected once a week, for 9 consecutive weeks. The nine Oregon graphs, which we name as *Oregon-A* through *Oregon-I*, correspond to individual weeks.
- **Gnutella Peer-to-Peer Networks** Gnutella graphs are peer-to-peer (P2P) connectivity networks collected daily, over consecutive days. The three Gnutella graphs, named *P2P-A* through *P2P-C*, correspond to individual days.
- **OpenFlights** and **US Airports** These are two networks of flights; one between US airports, and another between airports of the world in 2010, which we name as *Flight* and *Airport*, respectively.

## 5.3 Evaluation based on robustness improvement on all robustness measures

We start with employing EdgeRewire and all the existing methods as described in §5.1 on all of our real-world graphs and measure the improvement in robustness for all the six measures that we consider. Table 3 shows the robustness improvement percentages of our *EdgeRewire* versus the best performing baseline for each respective adjacency-spectrum based measure and Laplacian-spectrum based measure after 100 edge-rewirings and 40 edge-rewirings, respectively. It is clear that our *EdgeRewire* outperforms the best baseline by a very large margin for all the measures and on all of the datasets.

We remark that we are the first to introduce a degree-preserving rewiring framework/algorithm for spectral gap, natural connectivity, effective resistance, and number of spanning trees. As such, there is no existing method customized for those measures. One the other hand, there are degree-preserving algorithms such as those by Van Mieghem *et al.* [40] for spectral radius $\lambda_1$ and algebraic connectivity $\mu_2$. Therefore, in the following, we provide more in depth evaluation of performance on these two measures. For the rest of the text, we denote $\lambda_1$ and $\mu_2$ as $\lambda$ and $\mu$ for simplicity, and refer to EdgeRewire that optimizes the respective measures as *EdgeRewire*-$\lambda$ and *EdgeRewire*-$\mu$.

## 5.4 Evaluation based on robustness improvement on $\lambda$ and $\mu$

As mentioned in §5.1, we also consider a variant of each baseline that ensures improvement at every rewiring step. Correspondingly, we introduce AssortativeM (M, for modified), Smart-$\lambda$, and Random-$\lambda$ to guarantee that they increase $\lambda$ at every rewiring step. Moreover, we introduce DisassortativeM, Smart-$\mu$, and Random-$\mu$ for improving $\mu$ at every rewiring step. Van Mieghem *et al.* [40] show that Assortative improves $\lambda$ but decreases $\mu$, whereas Disassortative improves $\mu$ while decreasing $\lambda$. Therefore, we do not consider the opposite variants of them (i.e., it does not make sense to consider Assortative-$\mu$ and Disassortative-$\lambda$).

**Table 3** Edge-rewiring performances: robustness improvement percentages (higher is better) on all our real-world graphs after $k = 100$ and $k = 40$ rewirings for adjacency-spectrum based measures and Laplacian-spectrum based measures, respectively. $\lambda_1$: spectral radius, $\lambda_1 - \lambda_2$: spectral gap, N: natural connectivity, $\mu_2$: algebraic connectivity, R: effective resistance, and S: number of spanning trees. Values in non-italic for EdgeRewire and values in italic in parentheses for the best performing baseline (see §5.1) for each respective measure demonstrate the superiority of EdgeRewire over existing approaches.

| **Measures** | *O-A* | *O-B* | *O-C* | *O-D* | *O-E* | *O-F* |
|---|---|---|---|---|---|---|
| $\lambda_1$ | 12.41 | 11.71 | 8.89 | 7.65 | 6.72 | 7.52 |
| *(best baseline)* | *(8.05)* | *(5.38)* | *(3.58)* | *(2.54)* | *(1.92)* | *(1.89)* |
| $\lambda_1 - \lambda_2$ | 39.57 | 30.64 | 29.43 | 31.37 | 19.70 | 23.21 |
| | *(36.57)* | *(27.30)* | *(19.30)* | *(22.44)* | *(10.89)* | *(11.31)* |
| N | 14.86 | 13.28 | 9.84 | 8.45 | 7.34 | 8.17 |
| | *(9.77)* | *(6.07)* | *(3.75)* | *(2.78)* | *(2.37)* | *(2.20)* |
| $\mu_2$ | 95.90 | 61.09 | 53.33 | 58.01 | 55.37 | 299.16 |
| | *(43.72)* | *(12.37)* | *(1.39)* | *(23.19)* | *(9.88)* | *(42.68)* |
| R | 37.18 | 36.75 | 34.05 | 35.53 | 35.76 | 48.18 |
| | *(10.18)* | *(13.77)* | *(4.20)* | *(5.25)* | *(4.26)* | *(9.34)* |
| S | 7901.53 | 3649.20 | 858.62 | 997.89 | 4024.50 | 5157.66 |
| | *(2409.75)* | *(1618.57)* | *(283.79)* | *(551.54)* | *(146.27)* | *(413.05)* |

| **Measures** | *O-G* | *O-H* | *O-I* | *P-A* | *P-B* | *P-C* |
|---|---|---|---|---|---|---|
| $\lambda_1$ | 7.21 | 6.35 | 5.76 | 12.61 | 12.66 | 21.53 |
| *(best baseline)* | *(1.85)* | *(1.20)* | *(0.93)* | *(0.30)* | *(0.20)* | *(0.11)* |
| $\lambda_1 - \lambda_2$ | 23.00 | 19.67 | 17.47 | 49.15 | 53.65 | 114.59 |
| | *(9.96)* | *(5.96)* | *(4.96)* | *(3.77)* | *(4.04)* | *(2.85)* |
| N | 7.73 | 6.76 | 6.10 | 14.32 | 14.38 | 25.10 |
| | *(2.03)* | *(1.33)* | *(0.98)* | *(0.35)* | *(0.27)* | *(0.11)* |
| $\mu_2$ | 86.01 | 51.35 | 27.57 | 185.59 | 119.71 | 69.83 |
| | *(14.02)* | *(7.71)* | *(6.13)* | *(0.00)* | *(0.01)* | *(0.00)* |
| R | 41.35 | 42.83 | 31.43 | 37.18 | 10.64 | 6.88 |
| | *(3.84)* | *(3.62)* | *(2.98)* | *(1.35)* | *(0.18)* | *(0.18)* |
| S | 14898.57 | 4028.65 | 2398.93 | 18282.71 | 43580.95 | 92550.46 |
| | *(229.69)* | *(84.85)* | *(63.59)* | *(36.44)* | *(22.13)* | *(11.88)* |

| **Measures** | *Airport* | *Flight* |
|---|---|---|
| $\lambda_1$ | 1.28 | 3.53 |
| *(best heuristic)* | *(0.26)* | *(0.42)* |
| $\lambda_1 - \lambda_2$ | 2.13 | 9.59 |
| | *(1.70)* | *(4.85)* |
| N | 1.32 | 3.73 |
| | *(0.25)* | *(0.46)* |
| $\mu_2$ | 17.33 | 140.44 |
| | *(12.92)* | *(4.45)* |
| R | 42.92 | 42.47 |
| | *(3.40)* | *(4.85)* |
| S | *119051.84* | *591652.13* |
| | *(201.48)* | *(538.66)* |

To show that *EdgeRewire*-$\lambda$ and *EdgeRewire*-$\mu$ are effective in improving $\lambda$ and $\mu$, respectively, we compare the graphs rewired by our methods to graphs rewired by different heuristics. In particular, we record the $\lambda$ and $\mu$ values of the rewired graphs after each successive edge-rewiring. Figure 3 shows the $\lambda$ values for four example graphs using *EdgeRewire*-$\lambda$ and the corresponding heuristics where we perform the rewiring procedure up to 500 times. Figure 4 shows the $\mu$ values of the rewired graphs using *EdgeRewire*-$\mu$ and the corresponding heuristics where we repeat the procedure 200 times. Clearly, our methods obtain much higher $\lambda$ and

**Table 4** Edge-rewiring ($\lambda$) performances: $\lambda$ value (higher is better) after $k = 500$ rewirings.

| **Methods** | *O-A* | *O-B* | *O-C* | *O-D* | *O-E* | *O-F* | *O-G* | *O-H* |
|---|---|---|---|---|---|---|---|---|
| *EdgeRewire*-$\lambda$ | **24.38** | **37.05** | **43.50** | **41.94** | **48.49** | **52.33** | **61.38** | **67.28** |
| Assortative | 23.51 | 32.88 | 38.99 | 38.62 | 43.38 | 45.84 | 54.17 | 59.37 |
| AssortativeM | 24.04 | 33.53 | 39.45 | 39.17 | 43.87 | 46.41 | 54.81 | 59.96 |
| Smart | 20.86 | 29.59 | 36.20 | 36.25 | 40.90 | 43.49 | 51.67 | 57.18 |
| Smart-$\lambda$ | 21.07 | 30.26 | 37.04 | 37.55 | 42.44 | 45.11 | 53.39 | 58.67 |
| Random | 21.18 | 29.61 | 36.00 | 36.25 | 41.01 | 43.75 | 52.00 | 57.40 |
| Random-$\lambda$ | 23.28 | 31.85 | 38.05 | 38.50 | 42.85 | 45.56 | 53.78 | 58.98 |

| **Methods** | *O-I* | *P-A* | *P-B* | *P-C* | *Airport* | *Flight* |
|---|---|---|---|---|---|---|
| *EdgeRewire*-$\lambda$ | **71.93** | **42.56** | **43.28** | **35.06** | **117.69** | **71.49** |
| Assortative | 63.96 | 28.07 | 28.12 | 22.14 | 112.79 | 63.78 |
| AssortativeM | 64.34 | 28.83 | 28.75 | 22.49 | 113.75 | 64.36 |
| Smart | 61.98 | 27.71 | 27.90 | 21.96 | 110.89 | 60.62 |
| Smart-$\lambda$ | 63.22 | 28.43 | 28.48 | 22.44 | 112.62 | 63.28 |
| Random | 62.12 | 27.64 | 27.81 | 21.94 | 110.72 | 61.88 |
| Random-$\lambda$ | 63.52 | 28.65 | 28.62 | 22.45 | 113.14 | 63.96 |

**Table 5** Edge-Rewiring ($\mu$) performances: $\mu$ value (higher is better) after $k = 200$ rewirings. $\mu = 0.00$ implies that the graph becomes disconnected within $k$ rewirings.

| **Methods** | *O-A* | *O-B* | *O-C* | *O-D* | *O-E* | *O-F* | *O-G* | *O-H* |
|---|---|---|---|---|---|---|---|---|
| *EdgeRewire*-$\mu$ | **0.55** | **0.34** | **0.33** | **0.22** | **0.21** | **0.15** | **0.17** | **0.14** |
| Disassortative | 0.37 | 0.20 | 0.15 | 0.10 | 0.10 | 0.04 | 0.09 | 0.06 |
| DisassortativeM | 0.45 | 0.26 | 0.16 | 0.12 | 0.12 | 0.06 | 0.10 | 0.06 |
| Smart | 0.00 | 0.03 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Smart-$\mu$ | 0.26 | 0.13 | 0.14 | 0.09 | 0.09 | 0.03 | 0.07 | 0.06 |
| Random | 0.00 | 0.00 | 0.00 | 0.00 | 0.02 | 0.00 | 0.00 | 0.00 |
| Random-$\mu$ | 0.36 | 0.22 | 0.15 | 0.11 | 0.09 | 0.04 | 0.08 | 0.06 |

| **Methods** | *O-I* | *P-A* | *P-B* | *P-C* | *Airport* | *Flight* |
|---|---|---|---|---|---|---|
| *EdgeRewire*-$\mu$ | **0.13** | **0.32** | **0.22** | **0.46** | **0.39** | **0.20** |
| Disassortative | 0.08 | 0.10 | 0.07 | 0.19 | 0.25 | 0.05 |
| DisassortativeM | 0.08 | 0.09 | 0.07 | 0.18 | 0.25 | 0.05 |
| Smart | 0.00 | 0.04 | 0.05 | 0.13 | 0.19 | 0.00 |
| Smart-$\mu$ | 0.07 | 0.07 | 0.07 | 0.20 | 0.27 | 0.05 |
| Random | 0.02 | 0.04 | 0.04 | 0.11 | 0.20 | 0.04 |
| Random-$\mu$ | 0.08 | 0.11 | 0.10 | 0.18 | 0.25 | 0.05 |

$\mu$ values than other heuristics. While the plots are for some selected datasets, the plots for other datasets are very similar. Due to space consideration, we provide Table 4 and Table 5 to show the final $\lambda$ and $\mu$ values after 500 (for $\lambda$) and 200 (for $\mu$) edge-rewirings for all of the graphs.

In a separate experiment, we also studied the number of edge-rewirings required for the methods to achieve a certain percentage of improvement of the original $\lambda$ and $\mu$ values. We show these results for *Oregon-G* in Table 6 (similar pattern is observed for other datasets). We see that other heuristics require up to several orders of magnitude higher number of edge-rewirings than our methods to achieve the same improvement. Notice that the original versions of Smart and Random heuristics disconnect the graph before reaching the desired improvement for $\mu_2$.

## 5.5 Running time analysis of EdgeRewire

To study the running time of our EdgeRewire, we record (1) the total time (in seconds) to perform the $k^{th}$ edge-rewiring (here we refer to rewirings that

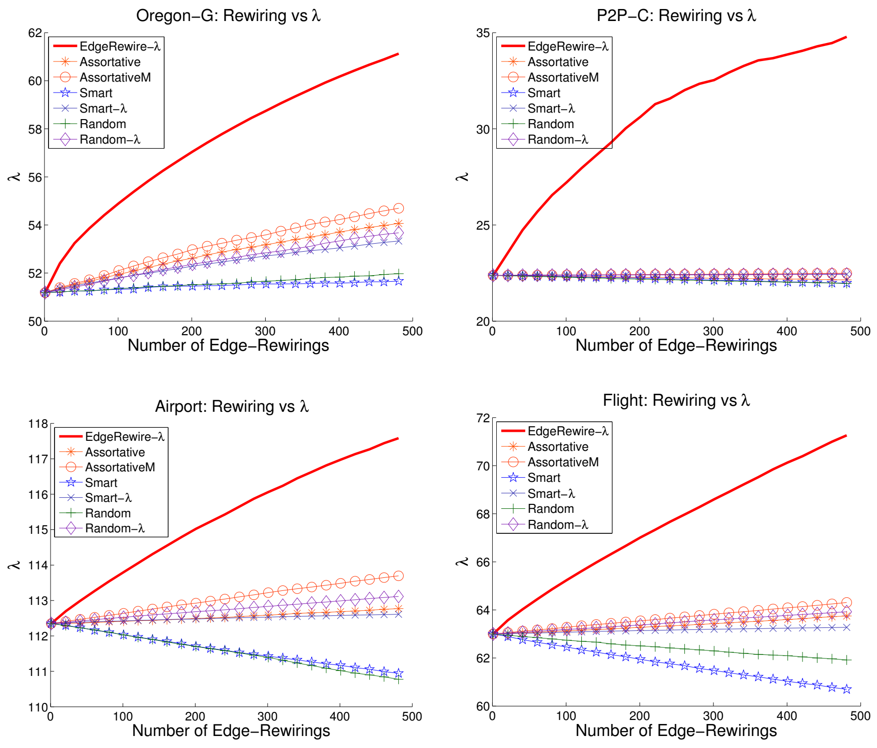

**Fig. 3** $\lambda$ value (higher is better) after $k$ edge-rewirings on *Oregon-G*, *P2P-C*, *Airport*, and *Flight*. Our method outperforms all heuristics by a large margin. (figures best in color)

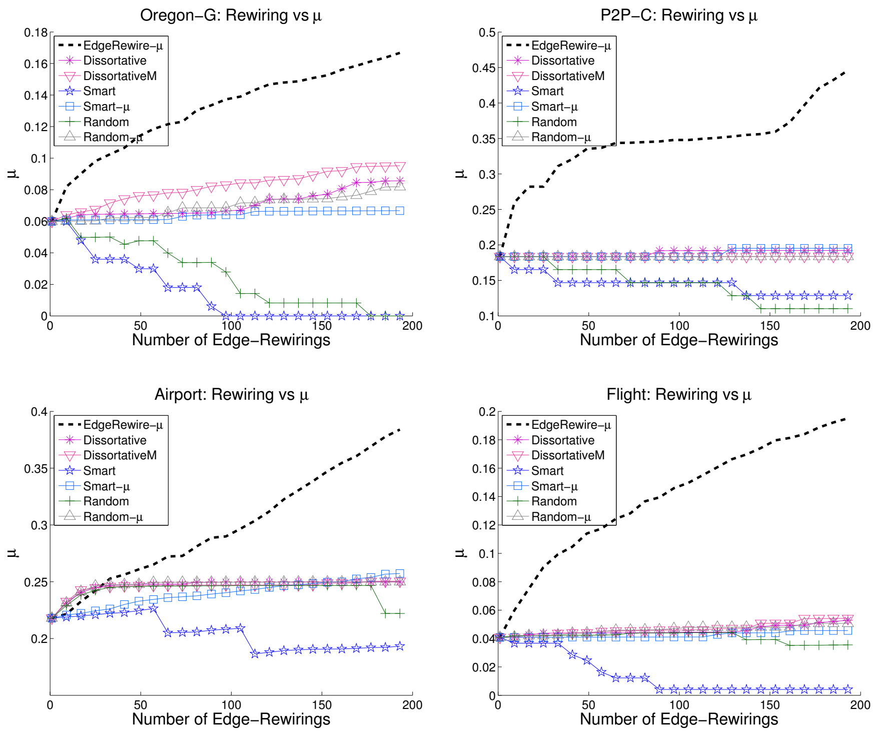

**Fig. 4** $\mu$ value (higher is better) after $k$ edge-rewirings on *Oregon-G*, *P2P-C*, *Airport*, and *Flight*. Our method outperforms all heuristics by a large margin. (figures best in color)

**Table 6** *Oregon-G*: Number of edge-rewirings required to increase $\lambda$ and $\mu$ by certain percentage (lower is better). * depicts disconnected graph, i.e., $\mu = 0$.

| | $\lambda$ | | | | $\mu$ | | | |
|---|---|---|---|---|---|---|---|---|
| **Methods** | 0.01% | 0.02% | 0.05% | 0.1% | 0.05% | 0.1% | 0.2% | 0.4% |
| *EdgeRewire*-($\lambda/\mu$) | **9** | **18** | **58** | **164** | **3** | **3** | **6** | **10** |
| (As/Disas)sortative | 66 | 139 | 411 | $>$ 500 | 14 | 94 | 121 | 166 |
| (As/Disas)sortativeM | 59 | 113 | 323 | $>$ 500 | 6 | 18 | 36 | 98 |
| Smart | $>$ 500 | $>$ 500 | $>$ 500 | $>$ 500 | * | * | * | * |
| Smart-($\lambda/\mu$) | 84 | 190 | 497 | $>$ 500 | 73 | 109 | $>$ 200 | $>$ 200 |
| Random | 330 | $>$ 500 | $>$ 500 | $>$ 500 | * | * | * | * |
| Random-($\lambda/\mu$) | 88 | 171 | $>$ 500 | $>$ 500 | 65 | 67 | 117 | >200 |

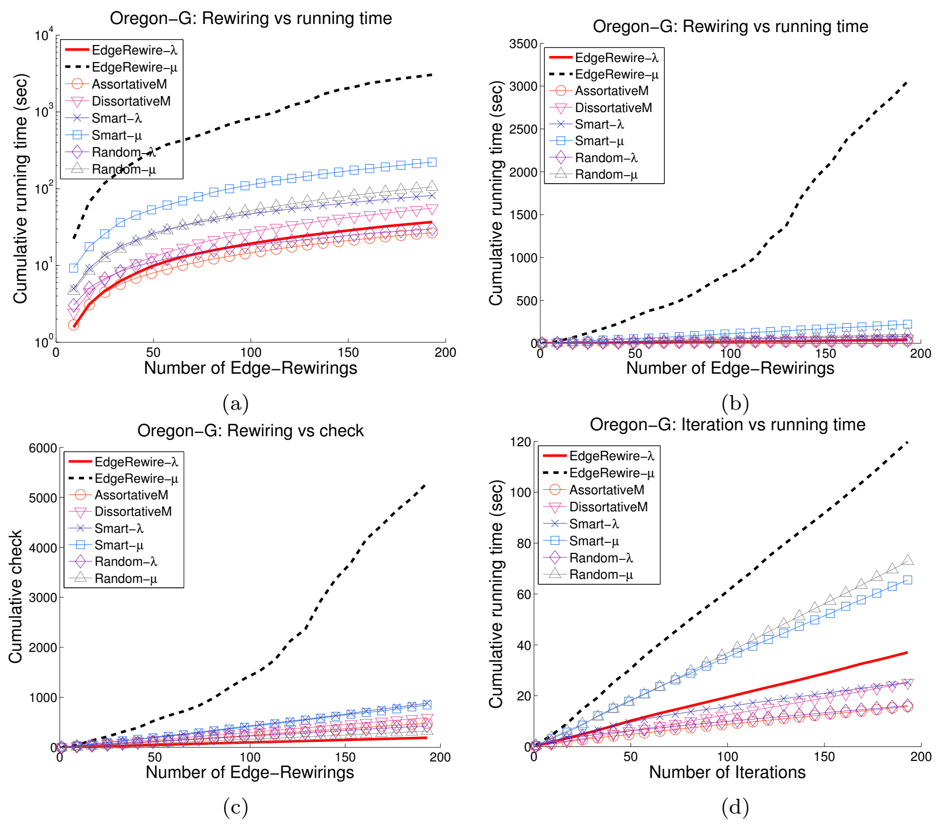

**Fig. 5** The plots on the top row show the total time in seconds (y-axis) to perform the $k^{th}$ successful edge-rewiring with improvement in robustness (x-axis) in log-y-axis scale (a) and linear scale (b). (c) Number of rewirings to inspect/check (y-axis) to perform the $k^{th}$ (successful) edge-rewiring (x-axis). (d) Total time in seconds (y-axis) to perform the $n^{th}$ inspection (x-axis). All of the plots are based on operations performed on *Oregon-G* and other datasets are similar. (figures best in color)

yield improvement in robustness as *successful*/committed rewirings), (2) the total number of candidate rewirings that are needed to be inspected (or "check"ed) to obtain the $k^{th}$ (successful) edge-rewiring, and (3) the total time to perform the inspections. Figure 5 shows that the running time of EdgeRewire grows linearly with respect to both the number of successful rewirings (5b) as well as the number of inspections (i.e., iterations) (5d). Notice that the total time of the methods (5b) is correlated with the number of checks/inspections that they make (5c) before committing a successful rewiring that yields improvement in robustness. However, each inspection takes constant time, and hence the linear growth of time by number of inspections (5d).

We note that *EdgeRewire*-$\mu$ takes the longest time to perform $k$ successful rewirings. However, we remark that it achieves significantly larger improvement in robustness with each rewiring compared to the other methods. We demonstrate this in Figure 6, which shows the total time (in seconds) taken by each method to achieve a certain percentage of robustness improvement for $\lambda$ (left) and $\mu$ (right). Notice that to achieve a certain level of robustness increase, *EdgeRewire* takes less time in general. Importantly, it performs significantly fewer edge-rewirings to achieve a particular level of increase (also visible through Figures 3 and 4). The latter is particularly critical for our task, as we often do not want to make too many changes to the network to increase its robustness to a desired level.

## 5.6 Evaluation based on attack tolerance

In addition to evaluating performance in improving a robustness value of interest, we also evaluate performance in achieving increased tolerance to attacks. In particular, we aim to quantify whether the graphs rewired by our framework are less vulnerable to attacks as compared to the graphs rewired by the other methods. Since the rewired graphs attain different $\lambda$ and $\mu$ values after a certain number of rewirings (i.e., budget), we measure the tolerance of a graph to an attack based on the fraction of the robustness value retained after each attack. Intuitively, if a graph is less vulnerable to an attack, then the attack would not change the value too much. Therefore, the graph that responds with the least marginal change on the value to a given attack is considered to be the most resilient, i.e., attack-tolerant.

Figure 7 demonstrates examples of how we determine which graph is the most resilient to attacks (we defer the specific definitions of what an attack constitutes to the next subsections). Specifically, if the fraction of $\lambda$ or $\mu$ retained after a series of attacks is consistently higher for a graph, then we conclude that the method that generated the graph is superior. In this particular instance, graphs rewired by our method are the most robust to various attacks (as defined and discussed below). We create Table 7 based on this principle and select the rewiring method that produces the graph that achieves the largest area under these resilience curves as shown in the figure. We use the graphs that have been rewired 500 times and 200 times for $\lambda$ and $\mu$ improvements, respectively.

### *5.6.1 Attacks to Decrease $\lambda$*

For graphs with $\lambda$ improvement, we consider two types of attacks. The first type of attacks are based on edge betweenness (B), and the second based on NetMelt [36] (N). In (B) attack, we select edges with the highest betweenness. In (N) attack, we select the edge $(p, r)$ with the highest value of $\mathbf{u_p u_r}$ where $\mathbf{u}$ is the principle eigenvector of $\lambda$ and $\mathbf{u_p}$ and $\mathbf{u_r}$ are the corresponding (node) values in the eigenvector. Table 7 (left) depicts the rewiring method that yields the most robust graph for different attacks on various real-world graphs. We see that our

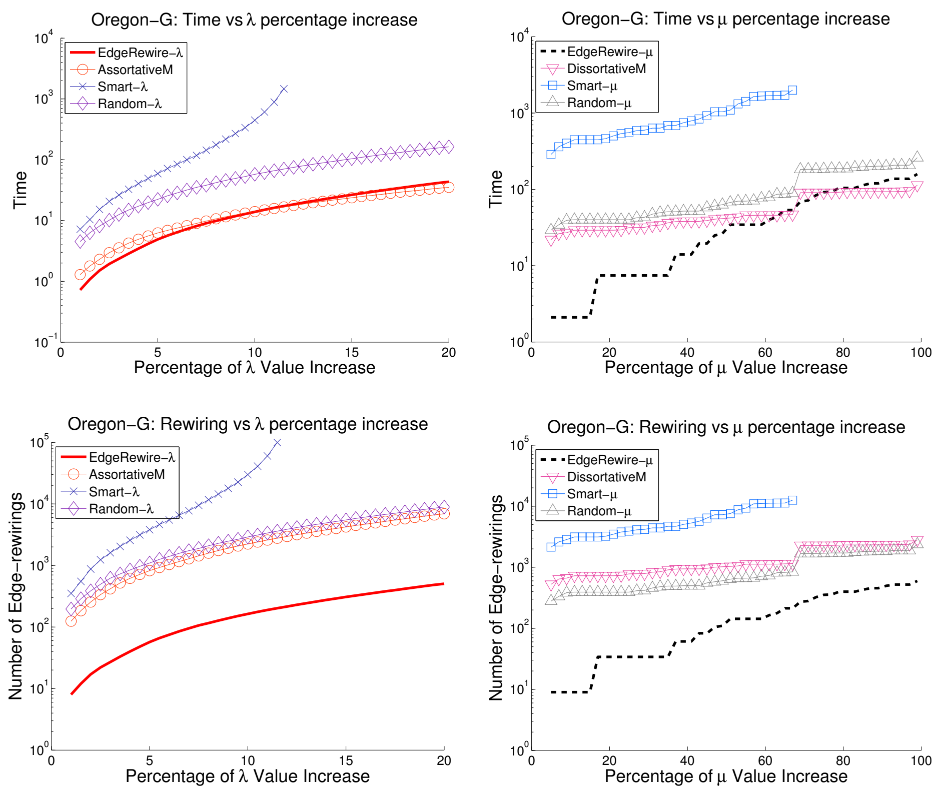

**Fig. 6** The plots on the top row show the total time in seconds (y-axis) to achieve a certain percentage of robustness improvement (x-axis) for $\lambda$ (left) and $\mu$ (right). The bottom plots show the number of edge-rewirings performed (y-axis) to achieve the certain level of robustness increase (x-axis) for $\lambda$ (left) and $\mu$ (right). All of the plots are based on *Oregon-G* and other datasets are similar. (figures best in color)

**Table 7 (left) Which $\lambda$-rewired graph is the most resistant to (100) attacks?** and **(right) Which $\mu$-rewired graph is the most resistant to (100) attacks?** The method that produces the most attack-tolerant graph is listed for both Betweenness based, and for NetMelt based attacks. EDGEREWIRE produces graphs that not only exhibit higher robustness with respect to a measure, but also with respect to external attacks.

| **Attacks:** | Betweenness | NetMelt | Betweenness | NetMelt |
|---|---|---|---|---|
| **Graphs** | $\lambda$-rewired | | $\mu$-rewired | |
| *Oregon-A* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | DisassortativeM |
| *Oregon-B* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | DisassortativeM |
| *Oregon-C* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Oregon-D* | *EdgeRewire-$\lambda$* | No Rewire | *EdgeRewire-$\mu$* | Smart-$\mu$ |
| *Oregon-E* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Oregon-F* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Oregon-G* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Oregon-H* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Oregon-I* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *P2P-A* | *EdgeRewire-$\lambda$* | Smart | *EdgeRewire-$\mu$* | DisassortativeM |
| *P2P-B* | *EdgeRewire-$\lambda$* | Smart-$\lambda$ | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *P2P-C* | *EdgeRewire-$\lambda$* | Random | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Flight* | AssortativeM | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |
| *Airport* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\lambda$* | *EdgeRewire-$\mu$* | *EdgeRewire-$\mu$* |

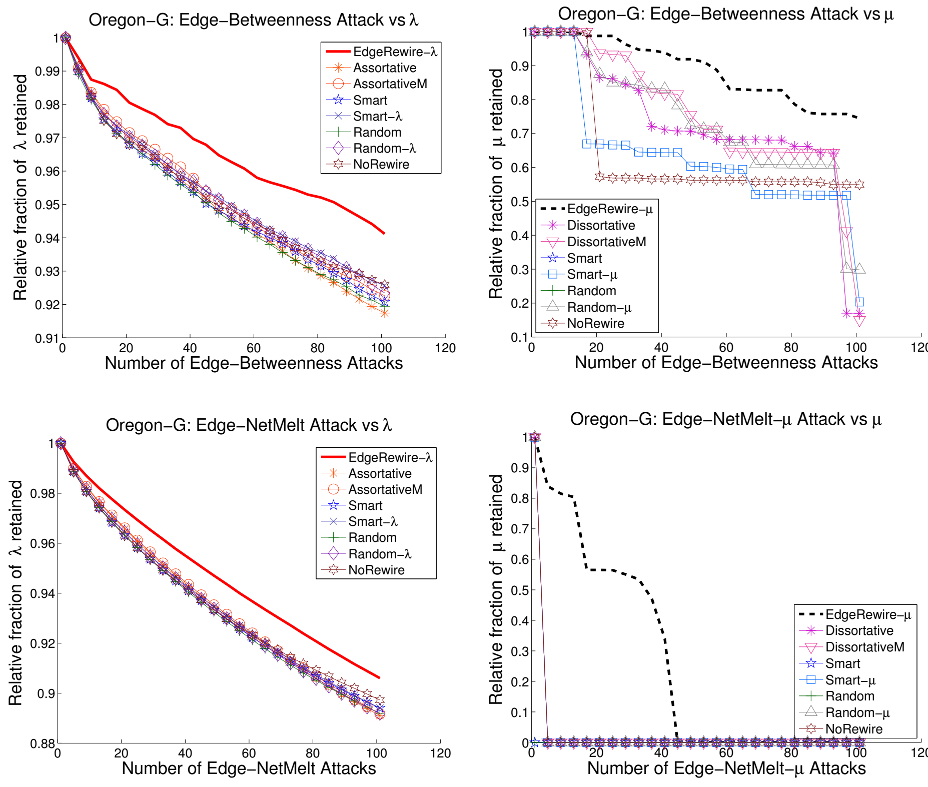

**Fig. 7** Relative fraction of $\lambda$ (left) and $\mu$ (right) value retained after $k$ attacks on the graphs rewired by *EdgeRewire*-$\lambda$ (left), *EdgeRewire*-$\mu$ (right), and corresponding heuristics on *Oregon-G*. Notice that graphs rewired by our methods are less vulnerable to different attacks compared to graphs rewired by other methods. (figures best in color)

*EdgeRewire*-$\lambda$ produces graphs that are most resilient to edge betweenness based attacks for all datasets but *Flight*. As for NetMelt based attacks, the graphs by *EdgeRewire*-$\lambda$ are also the most robust for 10 out of the 14 datasets. For the remaining four datasets, we do not observe any other particular method that is as consistently better as *EdgeRewire*-$\lambda$.

### *5.6.2 Attacks to Decrease $\mu$*

For graphs with $\mu$ improvement, we also consider attacks based on betweenness (B) and NetMelt-$\mu$ (N). NetMelt-$\mu$ is similar to the original NetMelt except that the importances of the edges are instead computed based on the eigenvectors of $\mu$. Table 7 (right) depicts the rewiring method that yields the most robust graph for different attacks on various real-world graphs. We observe that our *EdgeRewire*-$\mu$ produces graphs that are most robust to edge betweenness attacks for all of the datasets. The graphs by *EdgeRewire*-$\mu$ are also the most robust for 10 out of the 14 datasets with respect to the NetMelt-$\mu$ attacks (especially on the larger datasets within each category), where DisassortativeM appears to be the second best alternative. These experiments demonstrate that it is harder for edge betweenness and NetMelt-$\mu$ attacks to disconnect graphs rewired by our *EdgeRewire*-$\mu$.

## 5.7 Summary of Results

In this work, we conducted a long list of experiments to study the effectiveness, running time, and the attack tolerance achieved for our EdgeRewire on 14 real-world graphs from 3 different domains (AS router, P2P, flight networks), as compared to 8 baseline methods. To conclude, we provide a summary of our results.

We showed in Table 3 that our EdgeRewire outperforms the best baseline by a large margin for all the measures on all of the datasets under a given budget (i.e., number of rewirings). In particular, the significant improvement of our method over the baselines can be observed in Figure 3 (Table 4 for all datasets) and Figure 4 (Table 5 for all datasets) for $\lambda_1$ and $\mu_2$, respectively. We also studied the number of rewirings required by each method to achieve a certain percentage improvement on $\lambda_1$ and $\mu_2$. As shown in Table 6, the baseline heuristics require up to several orders of magnitude higher number of edge-rewirings than our methods to achieve the same robustness improvement. This is an important advantage of our method, as it is desirable to avoid making too many changes to the network to increase its robustness to a certain level. We also notice that the original versions of several baselines, Random and Smart, end up disconnecting the graph before being able to reach the desired improvement for $\mu_2$, whereas committing only the rewirings that improve the measure guarantees to retain connectivity.

Our study of the running time of EdgeRewire showed that the running time grows linearly with respect to the number of successful rewirings as well as the number of inspections (i.e., iterations) of the algorithm (Figure 5). We also studied the time required by each method to achieve a certain percentage improvement on $\lambda_1$ and $\mu_2$. As shown in Figure 6, the time required by EdgeRewire is significantly smaller than the baselines to achieve the same robustness improvement.

Our final set of experiments inspected the attack tolerance of graphs produced by the competing rewiring methods. As illustrated in Figure 7 and Table 7, we found that EdgeRewire produced graphs that are more resilient to external attacks. In particular, 13/14 and 14/14 graphs rewired by EdgeRewire were the most resilient against betweenness based attacks w.r.t. $\lambda_1$ and $\mu_2$, respectively. Similarly, 10/14 graphs rewired by EdgeRewire were more resistant against attacks based on NetMelt [36] w.r.t. both $\lambda_1$ and $\mu_2$.

All in all, our experiments demonstrate that EdgeRewire is superior to state-of-the-art baselines in producing graphs under a fixed budget that not only exhibit higher robustness as quantified by a spectral measure, but also with respect to external attacks. Moreover, its running time is linear with respect to the number of rewirings performed, where other methods require significantly larger number of rewirings (and hence time) to achieve the same robustness improvement.

## 6 Conclusion and Future Work

In this work we addressed the problem of modifying a graph's structure through degree-preserving edge rewirings so as to maximally improve its robustness under a specified budget, where robustness is quantified by spectral measures. We proposed a general framework called EdgeRewire that accommodates six different well-established spectral measures. Different from a vast literature of measure-independent heuristics, our approach leverages matrix perturbation theory to directly optimize a given measure. Experiments showed that EdgeRewire produces graphs with significantly higher robustness than several state-of-the art degree-preserving rewiring approaches. Moreover, our rewired graphs are, in general, less vulnerable to various attacks.

Optimizing the robustness of a network is an important problem for various applications, and our research sets off several future directions. First, it would be useful to devise algorithms like EdgeRewire, to maximally improve robustness when multiple measures are of interest to a given application. In other words, future research can focus on edge manipulation (add or rewire) algorithms that could improve multiple measures simultaneously. It would be interesting knowledge discovery to understand whether joint improvement of certain measures would be even possible. For example, Van Mieghem *et al.* [40] showed that their assortative algorithm improved $\lambda$ but decreased $\mu$, whereas their disassortative scheme improved $\mu$ while decreasing $\lambda$. This suggests a clash or "competition" between these two measures, and it is not directly obvious whether methods that can improve them both simultaneously can be designed. Another research direction is to devise other degree-preserving edge rewiring algorithms (heuristic or optimization-based) in order to improve those measures for which EdgeRewire is currently the only solution, including natural connectivity and effective resistance, and compare them to EdgeRewire. Alternatively, one could focus on improving the individual steps of EdgeRewire itself, either in speed or in accuracy of approximation. Finally, even though the optimal edge addition/deletion problem has been shown to be NP-hard for algebraic connectivity [28] and spectral radius [39], the computational complexity of the degree-preserving edge rewiring problem remains open.

## Acknowledgments

The authors thank the anonymous reviewers for their useful comments. This material is based upon work supported by the ARO Young Investigator Program under Contract No. W911NF-14-1-0029, NSF CAREER 1452425, IIS 1408287 and IIP1069147, a Facebook Faculty Gift, an R&D grant from Northrop Grumman Aerospace Systems, and Stony Brook University Office of Vice President for Research. Any conclusions expressed in this material are of the authors' and do not necessarily reflect the views, either expressed or implied, of the funding parties.

## References

1. L. Akoglu, M. McGlohon, and C. Faloutsos. Oddball: Spotting anomalies in weighted graphs. In *Proceedings of the 14th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining - Volume Part II*, PAKDD'10, pages 410–421, 2010.
2. R. Albert, H. Jeong, and A.-L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, July 2000.

3. J. Baras and P. Hovareshti. Efficient and robust communication topologies for distributed decision making in networked systems. In *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 2009 28th Chinese Control Conference*, CDC/CCC'09, pages 3751–3756, Dec 2009.
4. A. Beygelzimer, G. Grinstein, R. Linsker, and I. Rish. Improving network robustness by edge modification. *Physica A: Statistical Mechanics and its Applications*, 357(3–4):593 – 612, 2005.
5. A. E. Brouwer and W. H. Haemers. *Spectra of graphs.* Springer, New York, 2012.
6. F. Buekenhout and M. Parker. The number of nets of the regular convex polytopes in dimension <= 4. *Discrete Mathematics*, 186(1-3):69–94, 1998.
7. D. Chakrabarti, Y. Wang, C. Wang, J. Leskovec, and C. Faloutsos. Epidemic thresholds in real networks. *ACM Transactions on Information and System Security*, 10(4):1:1–1:26, Jan. 2008.
8. H. Chan, L. Akoglu, and H. Tong. Make it or break it: Manipulating robustness in large networks. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, SDM'14, pages 325–333, 2014.
9. H. Chan, S. Han, and L. Akoglu. Where graph topology matters: The robust subgraph problem. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, SDM'15, pages 10–18, 2015.
10. A. K. Chandra, P. Raghavan, W. L. Ruzzo, and R. Smolensky. The electrical resistance of a graph captures its commute and cover times. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 574–586, 1989.
11. D. M. Cvetković, M. Doob, and H. Sachs. *Spectra of Graphs: Theory and Application.* Academic Press, New York, 1980.
12. L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56:167–242, 2007.
13. W. Ellens and R. E. Kooij. Graph measures and network robustness. *CoRR*, abs/1311.5064:1–13, 2013.
14. W. Ellens, F. Spieksma, P. Van Mieghem, A. Jamakovic, and R. Kooij. Effective graph resistance. *Linear Algebra and its Applications*, 435(10):2491–2506, 2011.
15. E. Estrada. Network robustness to targeted attacks. the interplay of expansibility and degree distribution. *Physical Journal B - Complex Systems*, 52(4):563–574, 2006.
16. E. Estrada, N. Hatano, and M. Benzi. The physics of communicability in complex networks. *Physics Reports*, 514(3):89 – 119, 2012.
17. M. Faloutsos, P. Faloutsos, and C. Faloutsos. On power-law relationships of the internet topology. *SIGCOMM Computer Communication Review*, 29(4):251–262, Aug. 1999.
18. M. Fiedler. Algebraic Connectivity of Graphs. *Czechoslovak Mathematical Journal*, 23:298–305, 1973.
19. A. Ghosh, S. Boyd, and A. Saberi. Minimizing effective resistance of a graph. *SIAM Review*, 50(1):37–66, Feb. 2008.
20. P. Holme, B. J. Kim, C. N. Yoon, and S. K. Han. Attack vulnerability of complex networks. *Physical Review E*, 65(5):056109, 2002.

21. A. Jamakovic and P. Van Mieghem. On the robustness of complex networks by using the algebraic connectivity. In *Proceedings of the 7th International IFIP-TC6 Networking Conference on AdHoc and Sensor Networks, Wireless Networks, Next Generation Internet*, NETWORKING'08, pages 183–194, 2008.
22. D. J. Klein and M. Randić. Resistance distance. *Mathematical Chemistry*, 12(1):81–95, 1993.
23. V. Latora and M. Marchiori. A measure of centrality based on the network efficiency. *New Journal of Physics*, 9(188), 2007.
24. L. T. Le, T. Eliassi-Rad, and H. Tong. Met: A fast algorithm for minimizing propagation in large graphs with small eigen-gaps. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, SDM'15, pages 694–702, 2015.
25. V. H. P. Louzada, F. Daolio, H. J. Herrmann, and M. Tomassini. Smart rewiring for network robustness. *Journal of Complex networks*, 1(150-159), 2013.
26. F. D. Malliaros, V. Megalooikonomou, and C. Faloutsos. Fast robustness estimation in large social graphs: Communities and anomaly detection. In *Proceedings of the 2012 SIAM International Conference on Data Mining*, SDM'12, pages 942–953, 2012.
27. T. C. Matisziw and A. T. Murray. Modeling s-t path availability to support disaster vulnerability assessment of network infrastructure. *Computers and Operations Research*, 36(1):16–26, Jan. 2009.
28. D. Mosk-Aoyama. Maximum algebraic connectivity augmentation is np-hard. *Operations Research Letters*, 36(6):677–679, 2008.
29. M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67:026126, Feb 2003.
30. S. Saha, A. Adiga, B. A. Prakash, and A. K. S. Vullikanti. Approximation algorithms for reducing the spectral radius to control epidemic spread. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, SDM'15, pages 568–576, 2015.
31. S. Scellato, I. Leontiadis, C. Mascolo, P. Basu, and M. Zafer. Evaluating temporal robustness of mobile networks. *IEEE Transactions on Mobile Computing*, 12(1):105–117, Jan. 2013.
32. C. M. Schneider, A. A. Moreira, J. S. Andrade, S. Havlin, and H. J. Herrmann. Mitigation of malicious attacks on networks. *Proceedings of the National Academy of Sciences*, 108(10):3838–3841, Mar. 2011.
33. G. W. Stewart and J.-G. Sun. *Matrix Perturbation Theory.* Academic Press, New York, 1990.
34. F. Sun and M. A. Shayman. On pairwise connectivity of wireless multihop networks. *International Journal of Network Security*, 2(1/2):37–49, Mar. 2007.
35. A. Sydney, C. Scoglio, and D. Gruenbacher. Optimizing algebraic connectivity by edge rewiring. *Applied Mathematics and Computation*, 219(10):5465 – 5479, 2013.
36. H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 245–254, 2012.
37. H. Tong, B. A. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. H. Chau. On the vulnerability of large graphs. In *Proceedings of the 2010*

*IEEE International Conference on Data Mining*, ICDM '10, pages 1091–1096, 2010.

38. C. E. Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 608–617, 2008.
39. P. Van Mieghem, D. Stevanović, F. Kuipers, C. Li, R. van de Bovenkamp, D. Liu, and H. Wang. Decreasing the spectral radius of a graph by link removals. *Physical Review E*, 84:016101, Jul 2011.
40. P. Van Mieghem, H. Wang, X. Ge, S. Tang, and F. A. Kuipers. Influence of assortativity and degree-preserving rewiring on the spectra of networks. *The European Physical Journal B*, 76(4):643–652, 2010.
41. H. Wang and P. Van Mieghem. Algebraic connectivity optimization via link addition. In *Proceedings of the 3rd International Conference on Bio-Inspired Models of Network, Information and Computing Sytems*, BIONETICS '08, pages 22:1–22:8, 2008.
42. J. Wu, B. Mauricio, Y.-J. Tan, and H.-Z. Deng. Natural connectivity of complex networks. *Chinese Physics Letters*, 27(7):078902, 2010.
43. A. Zeng and W. Liu. Enhancing network robustness against malicious attacks. *Physical Review E*, 85:066130, 2012.