修士論文

ラプラシアン行列の第二最小固有値最大化が
ネットワーク結合の頑健性に与える影響

小川 泰司

主指導教員 林 幸雄

北陸先端科学技術大学院大学
先端科学技術研究科
（知識科学）

令和７年３月

Abstract

With the rapid advancement of computational capabilities, it has been known that complex networks such as social networks, communication networks, and biological systems exhibit Scale-free structures. Scale-free networks exhibit compactness and small-world properties, yet they suffer from extreme vulnerability to targeted attacks. Therefore, improving the robustness of networks with such structures is an urgent challenge.

The second smallest eigenvalue (μ_2) of the Laplacian matrix is known as an indicator of how difficult it is for a network to be bisected. The reason for this is that the diffusion equation in a network can be expressed as

$$\frac{d\mathbf{x}(t)}{dt} = -L\mathbf{x}(t),$$

where $L = D - A$ is the Laplacian matrix, and $\mathbf{x}(t)$ represents the information held by each node in the network. Additionally, D denotes a diagonal matrix $diag(k_1, k_2, \dots, k_i, \dots, k_N)$, where k_i is the degree of node i, and each element of an adjacency matrix A is defined by $A_{ij} = 1$ if nodes i and j are connected, otherwise $A_{ij} = 0$.The solution of the diffusion equation for $\mathbf{x}(t)$ is given by

$$\mathbf{x}(t) = \sum_{i=1}^{N} C_i e^{-\mu_i t}\mathbf{u}^{(i)} \approx C_1\mathbf{u}^{(1)} + C_2 e^{-\mu_2 t}\mathbf{u}^{(2)},$$

where C_i are constants determined by $\mathbf{x}(0)$, corresponding to the eigenvectors $\mathbf{u}^{(i)}$. The Laplacian matrix is a non-negative definite symmetric matrix, and since its eigenvalues always include 0, μ_2 is dominant in fast information transmission within the network.

However, the maximization of μ_2 is NP-hard. Therefore, as a heuristic method for maximizing μ_2, a method is proposed to add links between node pairs i and j where the absolute difference $|u_i - u_j|$ of Fiedler vector's (the second eigenvector's of Laplacian matrix) elements is maximized. Because $\Delta\mu_2 \approx \pm|u_i - u_j|$ is estimated, where the $\pm$ signs are corresponding to adding and removing links, respectively. However, a relation of μ_2 and the robustness of connectivity is unknown. Therefore, we discuss the impact of the robustness of connectivity in networks by maximizing the second smallest eigenvalue (μ_2) of the Laplacian matrix through various rewiring strategies.

First, for maximizing μ_2, we consider rewiring strategies in classical Erdős-Rényi (ER) random graphs with Poisson degree distributions and realistic Scale-Free networks with power-law degree distributions. In the rewiring process, new links are added between node pairs i and j, where the absolute difference of Fiedler vector's elements $|u_i - u_j|$ is maximized, while existing links are removed between

node pairs where $|u_i - u_j|$ is minimized. So, there are three different strategies: "all at once", "one by one", and our proposed "the intermediate" as follows.

Second, we emphasize that degree distributions are changeable with other topological structures such as degree-degree correlations and so on after rewiring. Therefore, to investigate the pure effect of changed degree distributions on the robustness of connectivity in eliminating these extra structures including chains, we consider the randomizations by using configuration model.

Third, we consider betweenness centrality attacks with strong destruction effect in investigating the robustness. Betweenness centrality is defined by $\sum_{s,t \in N} \frac{\sigma_{st}(v)}{\sigma_{st}}$, where $\sigma_{st}(v)$ represents the number of shortest paths between nodes s and t passing through node v. As a measure of robustness of connectivity, we use the robustness index $R \equiv \sum_{q=\frac{1}{N}}^{1} \left(\frac{S(q)}{N} \times \frac{1}{N} \right)$, where $S(q)$ denotes the size of largest connected component when $N \times q$ nodes are removed in descending order of betweenness centrality. The R value ranges from $1/N$ to $1/2$, and a larger value indicates greater robustness.

The results of the simulation conducted above are shown below.

- The intermediate strategy achieved the highest increase in μ_2 among the three approaches.
- The R-value and the variance of the degree distribution (σ^2) were strongly correlated, implying that an increase in μ_2 resulted in networks with lower σ^2.
- "All at once" generated star-like structures with large hubs, leading to high σ^2 and reduced robustness.
- The relationship between μ_2 and R was generally positive but not strictly proportional, suggesting that factors beyond μ_2 influence network robustness.

Furthermore, we will investigate such properties for other attacks, e.g., uniformly at random, degree-based, loop destruction, and so on.

目次

図 目 次

表 目 次

第1章　はじめに

1.1　研究背景

今世紀頃から、コンピュータの計算能力の飛躍的な向上によりSNS上の人間関係やWWW、電力網、航空路線網といった大規模なネットワークの解析が可能となった。これらを解析したところ、多くの現実のネットワークは、ごく一部のノードに大量のリンクが集中する高次数のノード（ハブ）と大多数の低次数ノードで構成される [1, 2, 3]、次数分布がべき乗則に従うことが判明した [4, 5]。一方、べき乗分布ではなく、身長、体重、成績、血圧などの人間の特徴や、鉱物の粒子サイズおよび砂粒サイズといった自然界の現象は、平均値（$\bar{x}$）を中心とした左右対称な山形を示す「正規分布（図 1.1 左)」が当てはまる。この分布においては分散（σ^2）が大きいほど山は低く広がり、小さいほど山は高く尖った形となるが、データの 99.7% は $\bar{x} \pm 3\sigma$ の範囲に収まる。つまり、平均から大きく外れる極端なハブは存在せずほとんどのデータは平均値の近傍に存在し、平均値が分布に特定のスケールを与えている。しかしながら、正規分布とは異なりべき乗則に従う「べき分布（図 1.1 右)」に当てはまる現象も数多く存在している。つまり、平均値から分布を特徴付けることができず、スケールが存在しない。べき分布に当てはまる例として、万有引力の法則やクーロンの法則における逆 2 乗則、地震の規模と発生頻度、生物学のスケーリング則、所得分布や商品販売数量など、様々な現象が挙げられている [15]。

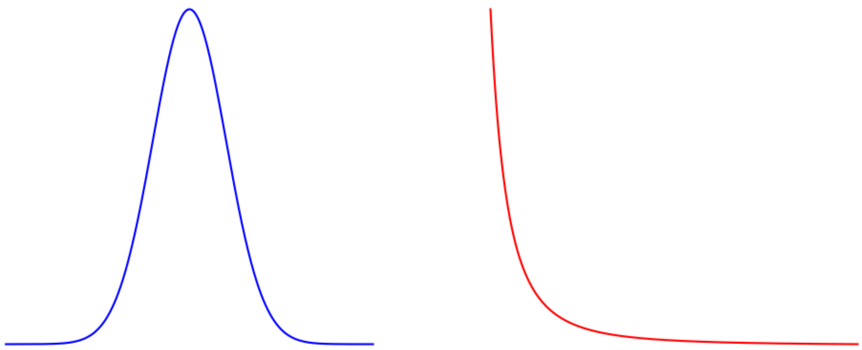

図 1.1: 正規分布（左）とべき分布（右）の形状

ここで、べき乗則は対象の尺度を変化させても性質が変化しない「スケール不変性」[15] を持つことから、このような特徴は「Scale-Free（SF）構造」[7] と名付けられ、この構造を持つネットワークは SF ネットワークと呼ばれる。現実の多くのネットワークでべき乗の次数分布をもつ SF ネットワークが生成される原理として、優先的選択 [7] が知られている。これは、利己的な便利さによって新規航空路線が多くの乗り継ぎ便を持つ羽田空港や関西国際空港といったハブ空港に集中することや、人脈形成において多くのつながりを持つ人物が選ばれやすいことなどに例えることができ、Barabási–Albert（BA）モデル [7] においては、次数に比例した確率でノードにリンクが追加されてネットワークが成長する。このような SF ネットワークは効率的である一方で、ネットワーク内で多くのつながりをもつハブが意図的な攻撃などにより極少数除去されるだけで、ネットワーク全体が分断されるという脆弱性を持つことが明らかになっている [8, 9]。現実のシステムに例えると、航空路線網や電力網のような社会インフラにおいて、ハブの機能停止が局所的な問題に留まらず、システム全体の崩壊に繋がる。実際に、2003 年 8 月、アメリカのオハイオ州にある主要な 3 つの電線路に樹木が接触し送電が停止しただけでアメリカ北西部からカナダにかけて大規模な停電が発生して、約 7000 億円の金融赤字だけでなく、交通麻痺により帰宅できず公園や路上で一晩を過ごす人々が多く出ている。しかしながら今日、「自己責任論」や過度な競争主義による効率性重視の結果、自分自身の利益を第一優先に考えた優先的選択が過剰にはたらくことでハブへのリンク集中が加速しており、システムの脆弱性はより深刻化している。例として、SNS でのフォロー関係や東京一極集中などが挙げられる。そのため、昨今頻発する大規模災害やテロの脅威が高まる現代において、我々の生活を支えるネットワークシステムの多くはこの重大な危険性を内包しており、このような脅威に対して結合機能を保つ強固な耐性を持つネットワークをどのように構築するのかは緊急の社会的課題である。

　これまで、悪意のある攻撃に対して頑健なネットワークを得る検討も進められている。例えば、次数分布がポアソン分布に従う古典的な Erdős–Rényi（ER）ランダムグラフ [10] と比べ、SF ネットワークは悪意のある攻撃に対してより脆弱であることが判明している [8] が、次数分布に着目した研究は SF ネットワークと ER ランダムグラフに関するピンポイントに留まっていた。しかしながら近年、SF ネットワークのべき乗分布からカットオフ付きべき乗、指数分布、Poisson 分布、さらに狭い次数分布の幅まで網羅的に連続変化する次数分布と頑健性の関係を調べた結果、次数分布の幅が小さいレギュラーグラフほど頑健性が高いことが発見されている [11]。また、ある固定した次数分布において、次数が近いノード同士がどの程度つながっているかを定量的に示す次数相関 [12] に着目して、強い正の次数相関を示す玉葱状構造 [13, 14] を逐次成長させると攻撃に対して高い頑健性を持つこと [15, 16, 17, 18] から、ネットワークを頑健にできる。但し、正の次数相関を強くするほど頑健性が向上するわけではないことに注意する [13]。また、ランダム疎グラフにおいて、攻撃に関するノード集合 Dismantling Set とループに関するノード

集合 Feedback Vertex Set(FVS) が等価であること [19] に着目して、最小 FVS のサイズを増やすループ強化リワイヤリング手法 [20] などが提案されている。このように、次数分布や次数相関、ループに着目して、攻撃に対するネットワークの最適耐性の構造を得る試みがこれまで行われてきた。

一方、スペクトラルグラフ理論ではネットワーク構造の情報を含んだラプラシアン行列の固有値と固有ベクトルが研究されている [21]。特に、拡散方程式の解と関連するラプラシアン行列の第二最小固有値 μ_2 は代数的連結性と呼ばれ、μ_2 が大きい程ネットワークの二分割が困難であることを示す。また、リンクの両端ノードにおける固有ベクトル成分の差は道路網におけるノード間の連結性の強弱を示す [22]。しかしながら、μ_2 とネットワーク結合の頑健性との関連性は未だ十分に分かっていない。

1.2　研究目的

本研究では、「ラプラシアンの第二最小固有値 μ_2 と結合の頑健性にはなんらかの関係性がある」と考え、μ_2 の最大化が頑健性にどのように寄与するかを明らかにする。現実的な SF ネットワークと古典的な ER ランダムグラフに対する、既存の μ_2 最大化のためのヒューリスティックなリワイヤリング手法と、新たなリワイヤリング手法を複数検討する。さらに、リワイヤリング後のネットワークにおいて、μ_2 と頑健性の相関関係を分析するとともに、μ_2 最大化によって生じる次数分布の変化を定量的に評価する。

1.3　本論の構成

本論文の構成は以下の通りである。

第二章：関連研究
本研究に関連する既存研究の説明および用語について述べる。ネットワーク生成モデル、ラプラシアン行列、ネットワークの頑健性について説明する。

第三章：提案手法
ネットワークのラプラシアン行列の第二最小固有値 μ_2 を最大化するリワイヤリングを行うにあたり、3 つの戦略を考える。

第四章：実験・評価
3 章で述べた 3 つの戦略に基づくリワイヤリングを現実的な SF ネットワークおよび古典的な ER ランダムグラフに適用して、μ_2 の変化、頑健性の変化、次数分布の変化について調べる。

第五章：おわりに
本研究の結果をまとめる。

第2章　関連研究

本稿では、ネットワーク科学における様々なシミュレーションに適用するアルゴリズムや指標、重要な用語について述べる。具体的な内容としては、

1. **ネットワーク生成モデル：**現実的なつながりを表現した SF ネットワークと比較対象としてよく用いられる古典的な ER ランダムグラフを適用する。それらのネットワークを生成するために用いた 2 つのネットワーク生成モデルについて述べる。

2. **ラプラシアン行列の第二最小固有値：**ラプラシアン行列の第二最小固有値 μ_2 を最大化するリワイヤリング手法における新たな戦略を考える。本稿では、まずラプラシアン行列と μ_2、リワイヤリングで用いる第二最小固有ベクトル $\mathbf{u}^{(2)}$ について解説する。また、μ_2 と $\mathbf{u}^{(2)}$ を分析することで得られるネットワークの性質や、μ_2 を最大化する既存のリンク追加法、さらには本研究で参考にしたリワイヤリング手法についても説明する。

3. **ネットワークの頑健性指標：**ある攻撃に対して、リワイヤリング前と後でネットワークの頑健性がどのように変化するのかを調べる為の頑健性指標を説明して、リワイヤリングによる次数分布の変化を測定する意義についても述べる。

2.1 ネットワーク生成モデル

2.1.1 Growing random Network モデル

Growing random Network（GN）モデル [23] は次数分布がべき乗に従うネットワークを生成する Barabási–Albert（BA）モデルを拡張したモデルである。BA モデルについては 1-1 でも触れたが、高次数ノードほど接続先に選ばれやすい優先的選択に基づきリンクを追加する生成モデルであり、具体的なアルゴリズムは以下の通りである。ここで、これから紹介するネットワーク生成モデルは毎時刻に 1 個の新ノードと m 本のリンクを追加して成長するモデルであり、初期ネットワークは次数 0 の孤立ノードが存在しない $m+1$ ノードからなる完全グラフとする。

Barabási–Albert モデル

1. $t = 1, 2, 3, \ldots$ の毎時刻に新ノードを 1 個追加、新ノードから既存ノードに m 本のリンクを張る。このとき、既存ノード i はその次数に比例した接続確率 k_i で選択される（優先的選択）。

2. 規定のノード数 N になるまで、1 を繰り返す。

ただし、BA モデルは次数分布がべき乗に従う SF ネットワークしか生成できない。そこで、新ノードから既存ノードにリンクをはる接続確率を変更して、様々な次数分布の形をもつネットワークを生成できるモデルとして、GN モデルと後述する逆優先的選択（IPA）モデルが提案されている。GN モデルでは接続確率を k_i^ν（$k_i^\nu \geq 0$）として、$\nu = 1$ では BA モデルと同様のべき乗分布に従う SF ネットワークが、$\nu = 0$ では指数分布に従うランダム接続のネットワークが、$0 < \nu < 1$ ではべき乗分布と指数分布の中間にあたるカットオフ付きべき乗分布に従うネットワークが生成でき、次数分布を連続的に変化させるネットワークが生成可能となる。

2.1.2 逆優先的選択モデル

逆優先的選択（IPA）モデル [24] は優先的選択とは逆に、接続確率を k_i^β（$k_i^\beta < 0$）に設定することで、低次数ノードほど接続先に選ばれやすいようにする。このとき、$\beta = -1$ では次数分布がポアソン分布に従う ER ランダムグラフに近く、$\beta = -100$ ではほとんどのノードの次数が m となるレギュラーグラフに漸近する。

本研究では、GN モデルと IPA モデルの両者によりネットワークを生成することで、SF ネットワークからレギュラーグラフに漸近したネットワークまで、次数分布を連続的に変化させたネットワークが生成できる（図 2.1 参照）。本研究では、接続確率を $\beta = 1, -1$ に設定し、SF ネットワークと ER ランダムグラフを生成する。ここでノードとリンクにより様々な関係性をネットワークで表現するグラフ理

論では、リンクの向きが存在するネットワーク（有向グラフ）やノード間での伝播にかかる時間やコストを再現するためにリンクに重みを与えたネットワーク（重み付きグラフ）など様々なネットワーク形態が考えられているが、本研究ではネットワーク内での繋がり方が及ぼす影響を測りたいため、扱うネットワークデータは全て無向グラフでリンクに重みのない単純グラフを対象としている。また、自分自身にリンクをはる自己ループや同じノード間に複数のリンクをはる多重リンクも禁止している。

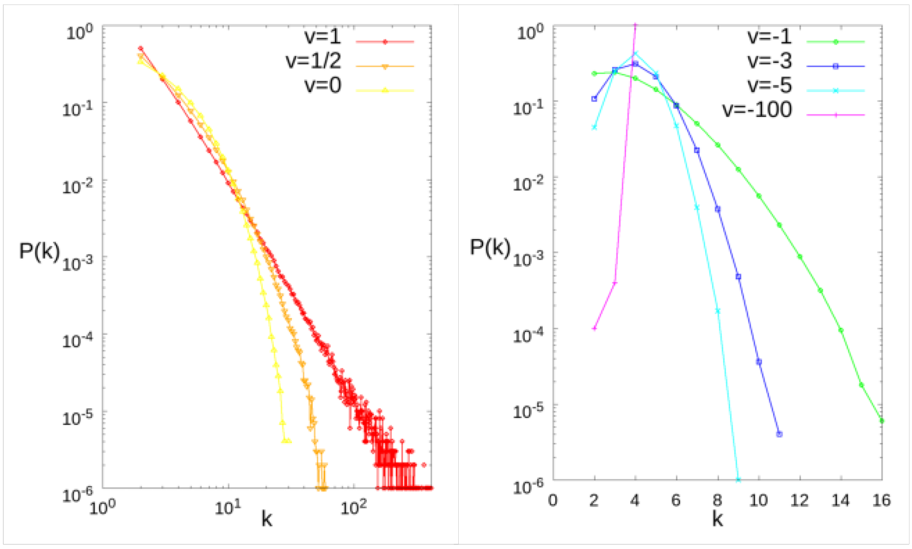

図 2.1: GN モデルと IPA モデルで生成したネットワークらの次数分布の変化。v を接続確率とし、左図（両対数プロット）にはべき乗分布（$v=1$）から指数分布 $v=0$）の、右図（片対数プロット）には $v=-1$ から $v=-100$ で生成されたネットワークでの結果を示す。

2.1.3 特殊な構造を除去するコンフィギュレーションモデル

前節で述べた新規ノードを追加するネットワーク生成手法や、リンク追加またはリワイヤリングにおいて、リンク生成過程に依存した特殊な構造が出現する場合がある。例として、IPA モデルでは β が大きい場合でネットワーク生成を行うとネットワークの直径が非常に大きい鎖状構造が発現することがわかっている [25]。そのため、次数分布の変化するリンク追加やリワイヤリング後のネットワークにおける攻撃耐性の変化をそのまま測ると、その変化の要因が特殊な構造によるものか次数分布の変化によるものかが不明となる。そこで、次数分布を保存したまま鎖状構造のような特殊な構造を除去して、ネットワークのつながりをランダム化するコンフィギュレーションモデル [26, 27] を適用することで、次数分布の影響のみを考えることができる。コンフィギュレーションの具体的な処理を図 2.2 に示す。まず、全てのリンクを切断し、ノードが自由端をもつ状態にする。次に、ランダムに選択した 2 つの自由端を再接続してリンクにする。このとき、同じノードペア間に複数のリンクができる多重リンクや自分自身にリンクをはる自己ループが起きないようにする。この自由端の再接続を繰り返し行い、ネットワークが連結するように全ての自由端を回収することで、次数分布はそのままで構造がランダム化されたネットワークが生成される。

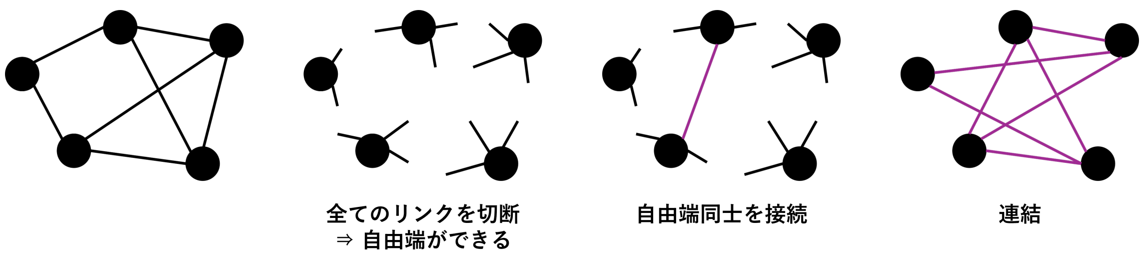

図 2.2: コンフィギュレーションモデルの処理過程

2.2 ラプラシアン行列の第二最小固有値

2.2.1 ラプラシアン行列

ネットワークを G, 総ノード数を N 個としたときにノードの集合を $V = \{0, 1, \ldots, i, j \ldots, N-1\}$、ノード $i - j$ 間のリンク集合を $E = \{e_{ij}\}$ とする。このとき隣接行列は

$$A_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

と定義される。ここで、（図 2.3 右）に示すネットワークにおける隣接行列を（図 2.3 左）に記述する。自己ループや多重リンクを禁止するネットワークでは必ず対角成分は 0 となり、ノード $i - j$ 間がリンクしている場合は (i, j) 成分が 1 となり、リンクしていない場合は (i, j) 成分が 0 となる。

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

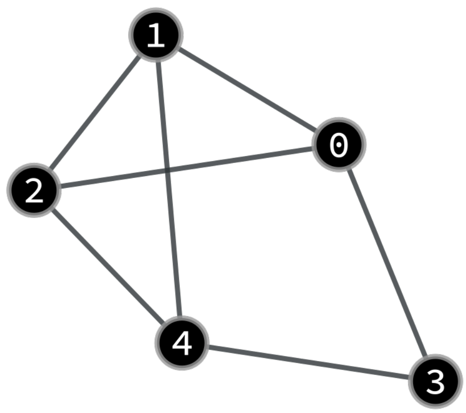

図 2.3: ノード 5 個からなるネットワーク（図 2.3 右）における隣接行列（図 2.3 左）

また、対角行列 D は式 (2.2) で定義される。ここで、d_i はノード i の次数を、D_{ij} は単位行列 I の成分を示す。

$$D_{ii} = d_i, \quad D_{ij} = 0, \quad i \neq j \tag{2.2}$$

すなわち、（図 2.4 左）に示す通り、対角成分 (i, i) はノード i の次数 d_i ある。

以上の隣接行列 A と対角行列 D を用いて、ラプラシアン行列 L[21] は次のように定義される。ここで、（図 2.5 左）に（図 2.5 右）に示すネットワークにおけるラプラシアン行列の行列成分を示す。これにより、ラプラシアン行列の i 行の成分の和が 0 であることがわかる。したがって、成分が全て 1 のベクトル $\mathbf{1} = (1, 1, \ldots, 1)^T$ は固有ベクトルとなり、$L\mathbf{1} = \mathbf{0} \times \mathbf{1}$ が成立する。ここで、^{T} はベクトルまたは行

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

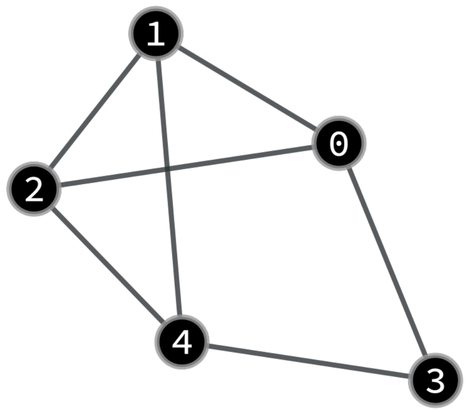

図 2.4: ノード 5 個からなるネットワーク（図 2.4 右）における対角行列（図 2.4 左）

列の転置を示す。すなわち、ラプラシアン行列には必ず零固有値が存在する。

$$L = D - A, \tag{2.3}$$

$$L = \begin{pmatrix} 3 & -1 & -1 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 \\ -1 & -1 & 3 & 0 & -1 \\ -1 & 0 & 0 & 2 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{pmatrix}$$

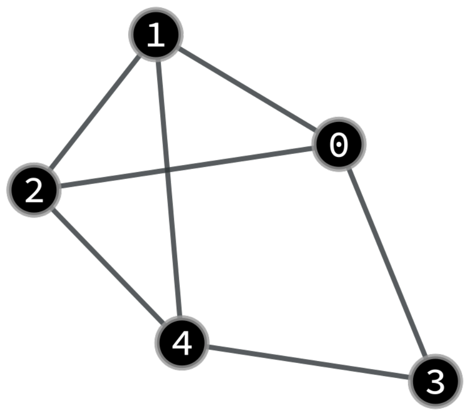

図 2.5: ノード 5 個からなるネットワーク（図 2.4 右）におけるラプラシアン行列（図 2.4 左）

　また、ラプラシアン行列の二次形式は以下の（式 2.4）となる。

$$\mathbf{x}^T L \mathbf{x} = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} (x_i - x_j)^2, \tag{2.4}$$

ここで、x_i および x_j はそれぞれベクトル $\mathbf{x}$ の i 成分と j 成分である。本研究では、隣接行列の成分 A_{ij} は 0 または正の値であるため、任意のベクトルに対するラプラシアン行列の 2 次形式は非負、すなわち、$\mathbf{x}^T L \mathbf{x} \geq 0$ となる。したがって、

ラプラシアン行列は非負定値対称行列であり、ラプラシアン行列の固有値は必ず非負の実数となる。また、ラプラシアン行列の固有値には必ず 0 が存在することから、ラプラシアン行列の固有値は

$$0 = \mu_{\mathbf{1}} \leq \mu_{\mathbf{2}} \leq \mu_{\mathbf{3}} \leq \cdots \leq \mu_{\mathbf{n}} \tag{2.5}$$

となる。ここで、μ_i はラプラシアン行列 L の昇順で i 番目の固有値を示す。

2.2.2 μ_2 を最大化する意義とその手法

$L = D - A$ と定義されるラプラシアン行列 L の第二最小固有値 μ_2 は、以下の拡散方程式

$$\frac{d\mathbf{x}(t)}{dt} = -L\mathbf{x}(t), \tag{2.6}$$

によって、ネットワーク内の高速な情報伝達において支配的であることが知られている。ここで、$\mathbf{x}(t)$ はネットワーク内の各ノードがもつ情報量を表す。その理由として、式 (2.6) の解は $\mu_{\mathbf{1}} = 0$ に注意して、

$$\mathbf{x}(t) = \sum_{i=1}^{N} C_i e^{-\mu_i t}\mathbf{u}^{(i)} = C_1\mathbf{u}^{(1)} + C_2 e^{-\mu_2 t}\mathbf{u}^{(2)} + \cdots + C_N e^{-\mu_N t}\mathbf{u}^{(N)}, \tag{2.7}$$

と求まる。また、式 (2.7) より t が十分に大きい時、第三項以降は式 (2.5) より無視できるため、

$$\mathbf{x}(t) \approx C_1\mathbf{u}^{(1)} + C_2 e^{-\mu_2 t}\mathbf{u}^{(2)}, \tag{2.8}$$

と近似できる。ここで、C_i は固有ベクトル $\mathbf{u}^{(i)}$ に対応する初期状態 $\mathbf{x}(0) = \sum_{i=1}^{N} C_i\mathbf{u}^{(i)}$ によって決まる定数であり、t によって $C_1\mathbf{u}^{(1)}$ は変化しない。これらのことから、μ_2 はネットワークの連結の強さを表す指標となり、代数的連結性（algebraic connectivity）とも呼ばれる [28]。

また、μ_2 に対応する第二最小固有ベクトル $\mathbf{u}^{(2)}$ はフィードラーベクトル（fiedler vector）[28] とも呼ばれる。ここで、図 2.6 にノード数 14 個からなるネットワークと各ノードにおける $\mathbf{u}^{(2)}$ 値を示す。ネットワークの可視化図から、ノード番号 6 と 7 間のリンクとノード番号 10 と 11 間のリンクがネットワークの連結性を保つ上で重要なリンクであり、このリンクを切断するとネットワークが分割されてしまうことが視覚的にわかる。このとき、各ノードの $\mathbf{u}^{(2)}$ 値をみてみると、ノード番号 6 と 7、ノード番号 10 と 11 の $\mathbf{u}^{(2)}$ 値の差が大きくなっていることがわかる。以上から、$\mathbf{u}^{(2)}$ 値の差はネットワークの連結性を保つ上でクリティカルであることが確認されている [22]。また、図 2.7 には、石川県の緊急輸送道路ネットワークを $\mathbf{u}^{(2)}$ の要素間の絶対差 $\left|u_i^{(2)} - u_j^{(2)}\right|$ で大きい順に順位付けした結果を示す。図 2.7 から、$\mathbf{u}^{(2)}$ の絶対差が高い道路（赤や黄色、緑の線）が遮断されると、道路が分断されることが視覚的にわかる。これらから、リワイヤリングにおけるネットワークの分割を防ぐ方法として、$\mathbf{u}^{(2)}$ の要素値に基づき、要素間の絶対差が大きいノードペアにリンクを補強し、要素間の絶対差が小さいノードペア間のリンクを削除する方法が考えられる（図 2.8 参照）。このとき、詳細は後述するが、リンクを追加するペアの $\left|u_i^{(2))} - u_j^{(2)}\right|$ が大きい程、μ_2 の微小変化に影響を与える。

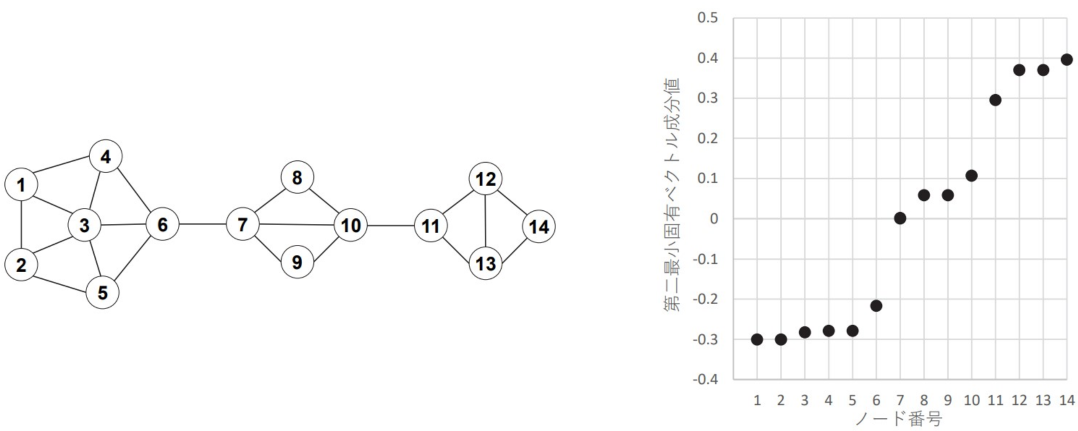

図 2.6: あるネットワーク（左）とその第二最小固有ベクトル $\mathbf{u}^{(2)}$ 値。出典 [22] から転載。

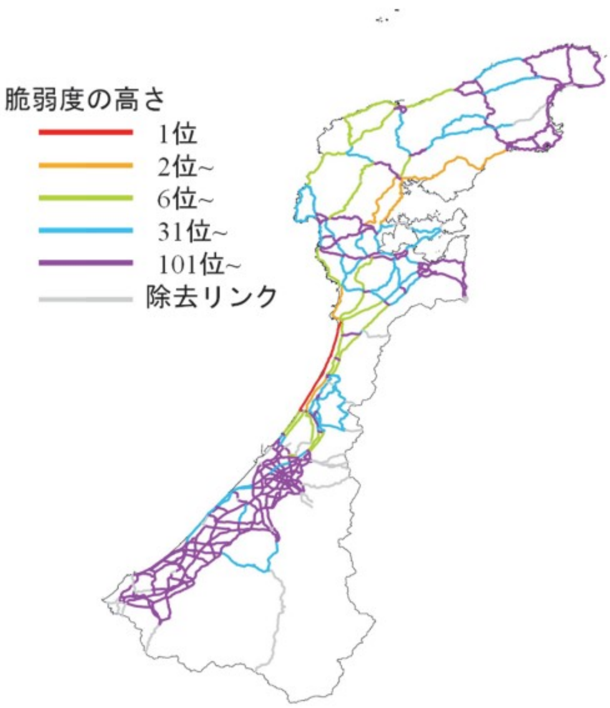

図 2.7: $\mathbf{u}^{(2)}$ の要素間の絶対差 $\left|u_i^{(2)} - u_j^{(2)}\right|$ による石川県の緊急輸送道路ネットワークにおける脆弱性評価。出典 [29] から転載。

① 第二最小固有ベクトル$u^{(2)}$の要素値に従い、未結合ノードi-j間をリンク、既結合ノードk-j間を除去してリワイヤリング

↓

② 1本または複数本のリワイヤリングで、$u^{(2)}$が更新される

↓

③ 更新された$u^{(2)}$で、①に戻って別のノードペアi-j間とk-l間をリワイヤリング

図 2.8: $\mathbf{u}^{(2)}$ に基づく、ネットワークの分割を防ぐリワイヤリング方法

μ_2 を大きくすることでネットワークの二分割を防いで連結性は強化されるが、ネットワーク内の全てのリンクを比較して μ_2 を最大化する最適なリンクを探索する問題は NP 困難 (NP-hard) な組み合わせ最適化問題である [30]。ここで、NP 困難とは計算複雑性理論において「複数の選択肢を同時に考える非決定過程において、ある答えが yes であると与えられたとき、その解が本当に正しいかどうかを多項式時間 ($O(N^k)$) で検証できる」NP（Non deterministic Polynomial）問題でありながら、一般的には効率的（多項式時間）に解を求めるアルゴリズムが存在しないと考えられている問題を指す [31]。具体的には、NP 困難な問題に対しては、解を探索する際に全ての可能な組み合わせを調べる必要がある場合が多く、これにかかる計算量は問題の規模に応じて指数的に増大する。そのため、実用的な規模の問題においては、現実的な時間内で最適解を得ることが極めて難しい。ラプラシアン行列の第二最小固有値 μ_2 を最大化する最適なリンクの探索も、この NP 困難な問題の一例である。この問題においては、ネットワーク内の全てのリンクの組み合わせを評価し、それぞれに対する μ_2 の変化を計算する必要があるが、この操作はネットワークの規模が大きくなるにつれて急速に計算量が増大する。特に、ノード数やリンク数が数千や数万に及ぶ大規模ネットワークでは、現実的な時間内に全てのリンクを比較することは事実上不可能である。そのため、NP 困難な問題に対しては、多くの場合で正確な最適解ではなく、近似解を求めるヒューリスティックなアルゴリズムが採用される。これらのアルゴリズムは、計算量を抑えつつも実用的な精度を保つことを目指して設計される。

そこで、μ_2 を最大化するヒューリスティックな方法として、ラプラシアン行列の第二最小固有ベクトル $\mathbf{u}^{(2)}$ の要素間の絶対差 $\left|u_i^{(2)} - u_j^{(2)}\right|$ が最大となるノードペア i-j 間にリンクを追加する手法 [32] や、そこから着想を得たリワイヤリングが提案されている [33]。以下に、その手順を示す。

Step1：ネットワークにおけるラプラシアン行列の μ_2 と第二固有ベクトル $\mathbf{u}^{(2)}$ を求める。

Step2：第二固有ベクトルの絶対差 $\left|u_i^{(2)} - u_j^{(2)}\right|$ が最大となるノードペア i-j 間と、絶対差 $\left|u_k^{(2)} - u_l^{(2)}\right|$ が最小となるノードペア間 k-l を取得する。

Step3：このとき、Step2 で取得した i-j 間にリンクがないかつ k-l 間にリンクがあるかつ k-l 間のリンクを削除したときにネットワークの接続性が保たれるかを確認する。一つでも条件を満たさない場合、他の候補を探索し、3 つ全ての条件を満たすまで繰り返す。

Step4：3 つ全ての条件を満たしたときリワイヤリングを行い、i-j 間にリンクを追加し k-l 間のリンクを削除する。

Step5：この操作をリワイヤリングが M' 回行われるまで繰り返す。

ヒューリスティックな方法として、$\left|u_i^{(2)} - u_j^{(2)}\right|$ が最大となるノードペア i-j 間にリンクを追加する手法、または $\left|u_k^{(2)} - u_l^{(2)}\right|$ が最小となるノードペア間のリンクを削除する手法が最適である理由として、μ_2 について

$$\Delta\mu_2 \approx \pm(u_i - u_j)^2, \tag{2.9}$$

が成り立ち [36]、リンクの微小変化に対する μ_2 の変化はそのリンクの両端ノードの $\mathbf{u}^{(2)}$ の成分の差の二乗であることを示す。ここで、$\pm$はリンク追加と削除に対応する。式 2.9は近似的に、$\left|u_i^{(2)} - u_j^{(2)}\right|$ が大きい場合はネットワークの連結性に大きく影響を与え、$\left|u_k^{(2)} - u_l^{(2)}\right|$ が小さい場合はほとんど影響を与えないことを意味する。

2.3　ネットワークの頑健性

2.3.1　攻撃に対する頑健性指標

攻撃に対してネットワークがどの程度の頑健性を有しているかを定量的に示す様々な指標が提案されている。ここでは本研究で用いる、頑健性指標 R は

$$R \equiv \sum_{q=\frac{1}{N}}^{1} \left(\frac{S(q)}{N} \times \frac{1}{N} \right), \tag{2.10}$$

で定義される [13]。ここで、N はネットワークのノード数、$q = \frac{1}{N}, \frac{2}{N}, \ldots, \frac{N-1}{N}, \frac{N}{N} = 1$ は攻撃により除去されたノード数の割合、$S(q)$ は $N \times q$ 個のノードが攻撃によって除去された時の最大連結成分に含まれるノード数を示す。つまり、R はノードを 1 つずつ順番に除去する度に求める $S(q)$ の平均値を表す。したがって、R 値は $1/N$ から $1/2$ までの範囲をとり、R 値が大きいほどそのネットワークが攻撃に対して頑健であることを示す。ここで、図 2.9 に例として、ネットワークへの攻撃による最大連結成分の変化を示す。曲線より下の面積が頑健性指標 R 値を示し、この面積が大きいほど頑健性が高いネットワークとなる。攻撃に対して脆弱なネットワークは、連結性にクリティカルなノードが除去されてしまうと、一気に分断されてしまうため、最大連結成分値が急降下する（図 2.9 の水色、青、緑で示したネットワーク）。逆に、攻撃に対して頑健なネットワークは 8 割程度ノードが除去された場合でも、残っているノードらは連結しているため、最大連結成分値はほぼ除去されたノード数分だけ変化して、ゆるやかに減少する（(図 2.9 の黄色、赤で示したネットワーク）。このとき、最大連結成分値が急激に減少する q 値をパーコレーション閾値と呼び、この値が高いほど攻撃によって分断されにくいネットワークであることを示す。

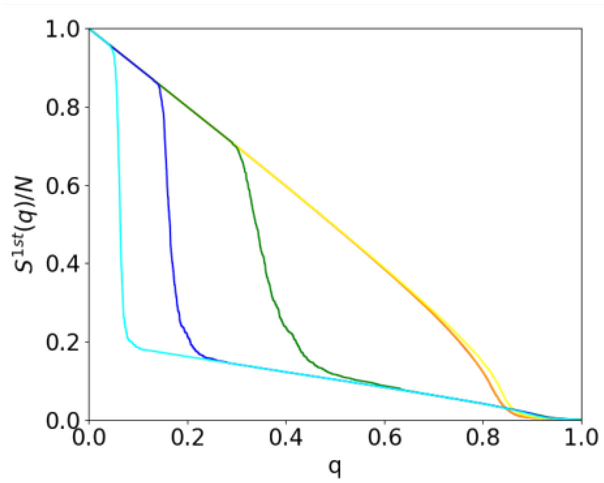

図 2.9: 攻撃による最大連結成分の変化の例

2.3.2 媒介中心性

ネットワークを攻撃する際、中心的な役割を担うノードを優先して除去していく方法が効率的にネットワークを分断すると考えられる。そのようなノードを見つけるために様々な中心性指標が提案されているが、本稿ではネットワークの最短経路において頻繁に出現するノードは中心性が高いとした媒介中心性を用いる。媒介中心性は

$$\sum_{s,t \in N} \frac{\sigma_{st}(v)}{\sigma_{st}}, \tag{2.11}$$

で定義される [34]。ここで、σ_{st} はノード s と t 間にある全ての最短経路数を表し、$\sigma_{st}(v)$ はそれらの最短経路の中で、ノード v を通る最短経路の数を示す。媒介中心性を指標とした攻撃 [35] は、ネットワーク内の情報伝播で頻繁に経由するノードが優先的に除去されるため（図 2.10 参照）、この攻撃に対して高い R 値をもつネットワークは頑健なネットワーク結合をもつ。

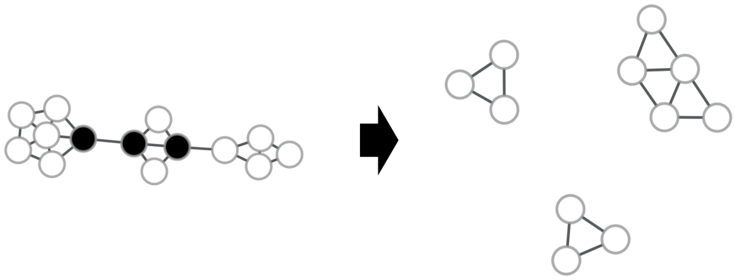

図 2.10: 媒介中心性攻撃の模式図。橋渡しの役割をもつノード（黒）が優先的に除去され、ネットワークの連結性が失われる。

2.3.3 次数分布の連続変化とネットワーク頑健性の関係性

近年、次数分布がネットワークの頑健性に深く関連していることが報告されている [36]。これまで、攻撃に対する頑健性についての研究はほとんど SF ネットワークと ER ランダムグラフに留まってピンポイントだったが、GN モデルと IPA モデルを用いて SF ネットワークのべき乗分布からカットオフ付きべき乗、指数分布、Poisson 分布、さらに狭い次数分布の幅まで連続変化する次数分布をもつネットワークを網羅的に生成して、次数分布と頑健性の関係を調べた結果、次数分布の幅が狭いレギュラーグラフほど頑健性が高いことがわかってきた [11]。そのため、次章以降でネットワークの μ_2 を最大化するリワイヤリングを行う際、前述した局所的構造を除去するコンフィギュレーションモデルを適用して、次数分布のみの変化が頑健性に与える影響を調べることで、μ_2 の変化や構造変化が頑健性に与える影響を明確にする。

第3章　提案手法

本章では、ネットワークのラプラシアン行列の第二最小固有値 μ_2 を最大化するリワイヤリングを行うにあたり、3つの戦略を考える。まず、μ_2 を最大化する方法として、Sydney らが提案したリワイヤリング手法 [33] に基づき、リンク追加と削除が行われる前のネットワークにおける μ_2 と $\mathbf{u}^{(2)}$ を求め、$\mathbf{u}^{(2)}$ の絶対差 $\left|u_i^{(2)} - u_j^{(2)}\right|$ の最大値と最小値に基づいてリンク追加と削除を M' 回だけ繰り返し行う（図3.1）。ここで、$\mathbf{u}^{(2)}$ は μ_2 に対応する第二最小固有ベクトルを表す。

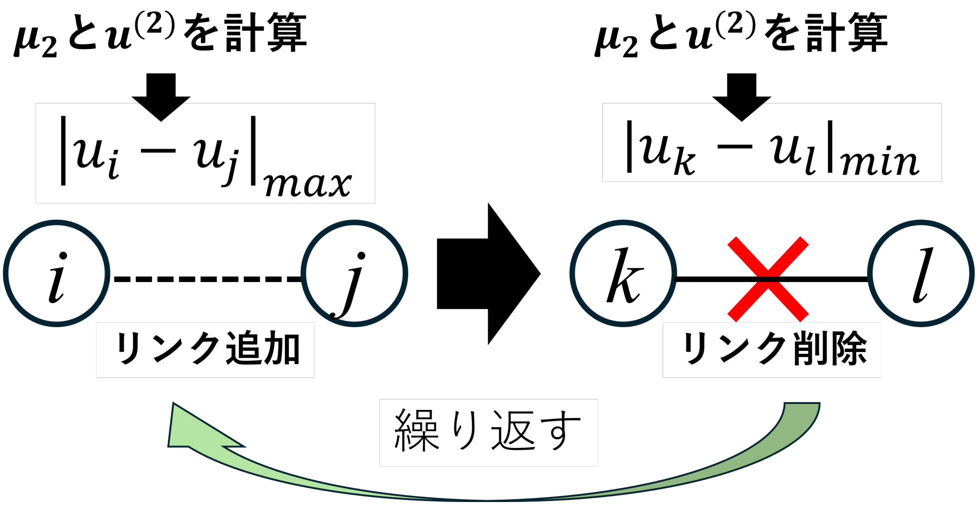

図 3.1: 本研究でのリワイヤリング手法

3.1 リワイヤリング戦略の提案

上記の従来法では、μ_2 を最大化するリワイヤリングにおいては 1 本ずつのリンク追加と削除を M' 回繰り返す、いわゆる「逐次戦略」だけが考えられてきた。しかしながら、リワイヤリングによる μ_2 の最大化を行う際にはいくつかの戦略が考えられる。本研究では逐次戦略に加え、新たに 2 つの戦略「一括戦略」と「中間戦略」を考え、3 つの戦略に基づくリワイヤリングを行う。

1. **逐次戦略：**従来行われてきたリワイヤリング戦略で、1 本ずつリンク追加とリンク削除する操作を M' 回繰り返す。この戦略に基づくリワイヤリングでは μ_2 と $\mathbf{u}^{(2)}$ を $2M'$ 回再計算する必要があるため、計算量が膨大になる。

2. **一括戦略：**初期状態のネットワークにおいて、リンクしていない全てのノードペアにおける $\left|u_i^{(2)} - u_j^{(2)}\right|$ を算出し、絶対差が大きい順に 1 本ずつリンク追加を行い合計 M' 本のリンクを一度に追加する。その後、μ_2 と $\mathbf{u}^{(2)}$ を再計算し、リンクしている全てのノードペアにおける $\left|u_k^{(2)} - u_l^{(2)}\right|$ を算出し、絶対差が小さい順に 1 本ずつリンク削除を行い合計 M' 本のリンクを一度に削除する。このとき、そのままリンク削除を行うとネットワークの連結性が失われる可能性があるため、1 本リンク削除を行う度にネットワークの最大連結成分が総ノード数と同値かを確認する。もし、連結性が失われていた場合は削除したリンクを張り直し、そのノードペア間リンクの絶対差の次に小さい絶対差をもつノードペア間のリンクを削除対象として、連結性が保持されるまでこの操作を繰り返す。

3. **中間戦略：**1 回のリワイヤリングにおいて複数の l 本のリンクを追加して、同じ数だけリンクを削除する。このリンク削除においても連結性が失われる可能性があるため、一括戦略で述べた最大連結成分を保持するための操作を行う。この過程はリンク追加と削除の前に μ_2 と $\mathbf{u}^{(2)}$ を再計算しながら c 回反復する。つまり、l は一度にリワイヤリングする本数、c は全リワイヤリング数 $M' = l \times c$ における反復回数を表す。

第4章　実験・評価

本章では、3章で述べた3つの戦略に基づくリワイヤリングを現実的なSFネットワークおよび古典的なERランダムグラフに適用して、以下に述べるシミュレーションをPythonのgraph-tool（付録A参照）を用いて実装して行う。

1. **3つの戦略に基づくリワイヤリングによる μ_2 変化の比較：**リワイヤリング過程におけるネットワークのラプラシアン行列の第二最小固有値 μ_2 を測定して、3つの戦略における μ_2 の変化を比較する。

2. **μ_2 とネットワーク結合の頑健性および次数分布間の関係性：**リワイヤリング後のネットワークにおける頑健性を評価するため、破壊力のある媒介中心性攻撃に対する頑健性指標 R と次数分布の分散 σ^2 を測定することで、μ_2 とネットワーク結合の頑健性および次数分布との関係を明らかにする。

シミュレーションで用いるSFネットワークとERランダムグラフはノード数 $N = 1000$、リンク数 $M = 1997$ とし、GNモデルとIPAモデルを組み合わせて生成した。ここで、$M = \binom{3}{2} + 2 \times (1000 - 3) = 1997$ となるのはノード数 $m + 1$ の完全グラフに対して、$t = 1, 2, 3, \ldots, 997$ の毎時刻に新ノードを1個追加して、新ノードから既存ノードに $m = 2$ 本のリンクを張る操作を $N = 1000$ になるまで繰り返すためである。また、リワイヤリング回数は $M' = 1000$ 回に設定、100サンプルの平均値を基に解析を行う。

リワイヤリングの際に、次数分布とそれ以外のネットワーク構造が変化するので、リワイヤリング後のネットワーク（以降の節ではSF：rewireおよびER：rewireと表記）の μ_2 値と R 値を測るだけではその変化の要因が次数分布変化なのか他の構造変化なのかがはっきりしない。そのため、リワイヤリング後のネットワークに対して次数分布を保存したまま構造をランダム化するコンフィギュレーションモデルを適用したネットワーク（以降の節ではSF：configおよびER：configと表記）における μ_2 値と R 値も求めて、次数分布のみの変化によるものとして比較する。

4.1 3つの戦略に基づくリワイヤリングによるμ_2変化の比較

図 4.1 に、3 つの戦略に基づく各リワイヤリング後の μ_2 値の分布を示す。赤の点線（SF:initial）と青の点線（ER：initial）はリワイヤリング前の SF ネットワークと ER ランダムグラフの 100 サンプル平均における μ_2 値をそれぞれ表す。また、図の凡例に示す「SF：rewire（ピンク）」と「ER：rewire（水色）」はリワイヤリング操作後のネットワークにおける μ_2 値の 100 サンプル平均、「SF：config（紫）」と「ER：config（緑）」はリワイヤリング後のネットワークにコンフィギュレーションモデルを適用したネットワークにおける μ_2 値の 100 サンプル平均を表す。横軸の「$l \times c$」において $l = 1$、$c = 1000$ が逐次戦略、$l = 1000$、$c = 1$ が一括戦略、それ以外のラベルが中間戦略を表す。

SF：rewire（ピンク）と ER：rewire（水色）の両方において、$l = 2$ で $c = 500$ の中間戦略が最も μ_2 値を向上させた。この傾向は、ER：config（緑）のネットワークにもみられるが、全ての戦略において ER：rewire（水色）と比べて μ_2 値が少し小さい。SF：config（紫）に対しては、どの戦略においても μ_2 がほぼ同じ値を示しており、SF：rewire（ピンク）と比べて μ_2 がはるかに小さいだけでなく、SF：initial（赤線）よりも小さくなっている。これらの結果から、μ_2 を最大化する戦略は一度に全てリワイヤリングする一括戦略でもなければ、大規模な計算を伴い一つずつリワイヤリングする逐次戦略でもないことが判明した。但し、μ_2 を最大化する問題は NP 困難であるため、厳密な最適解を求めることが現実的でなく、近似的な計算手法を用いている。つまり、$\left|u_i^{(2)} - u_j^{(2)}\right|$ の一次近似に基づいた、μ_2 の微小変化による最大化を図る。その結果、リワイヤリングで得られる解には一定の誤差が生じているため、これが要因として考えられる。また、凡例の rewire（ピンクと水色）と config（紫と緑）の μ_2 値の差から、次数分布以外の要因が存在することが伺える。

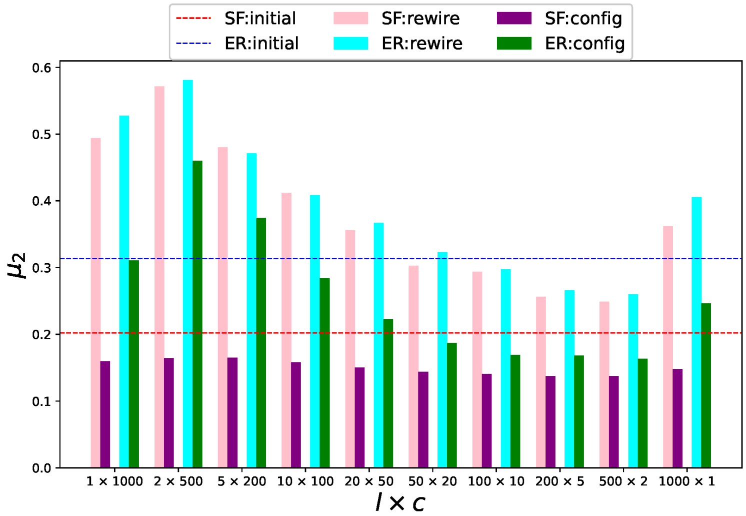

図 4.1: リワイヤリング後の μ_2 の分布

図 4.2 と図 4.3 に、それぞれ SF ネットワークと ER ランダムグラフにおける 3 つの戦略に基づくリワイヤリング回数に対する μ_2 の変化を示す。SF ネットワークでは、どの戦略においてもリワイヤリングが行われる度に μ_2 値が単調増加している。図 4.1 の SF：rewire（図 4.1 でのピンク）と SF：initial（図 4.1 での赤点線）における μ_2 値からも単調増加が示される。しかしながら、ER ランダムグラフでは、逐次戦略（図 4.3 での赤線）と $l = 2$ で $c = 500$ の中間戦略（図 4.3 でのピンク線)、一括戦略（図 4.3 での黒線）以外のリワイヤリングでは途中から μ_2 が低下している。また、図 4.1 の $l = 100$、$c = 10$ から $l = 500$ で $c = 2$ に対応する ER：rewire（図 4.1 での青）は ER：initial（図 4.1 での青点線）よりも小さい μ_2 値を示している。ER ランダムグラフにおいては、一度にリワイヤリングする本数が多く、反復回数の少ない中間戦略においては μ_2 値が減少し逆効果な場合もある。これらの挙動がなぜそうなったかについては不明であるが、μ_2 を最大化する問題が NP 困難であり、厳密な解を得られない一次近似による計算過程が要因であると考えられる。

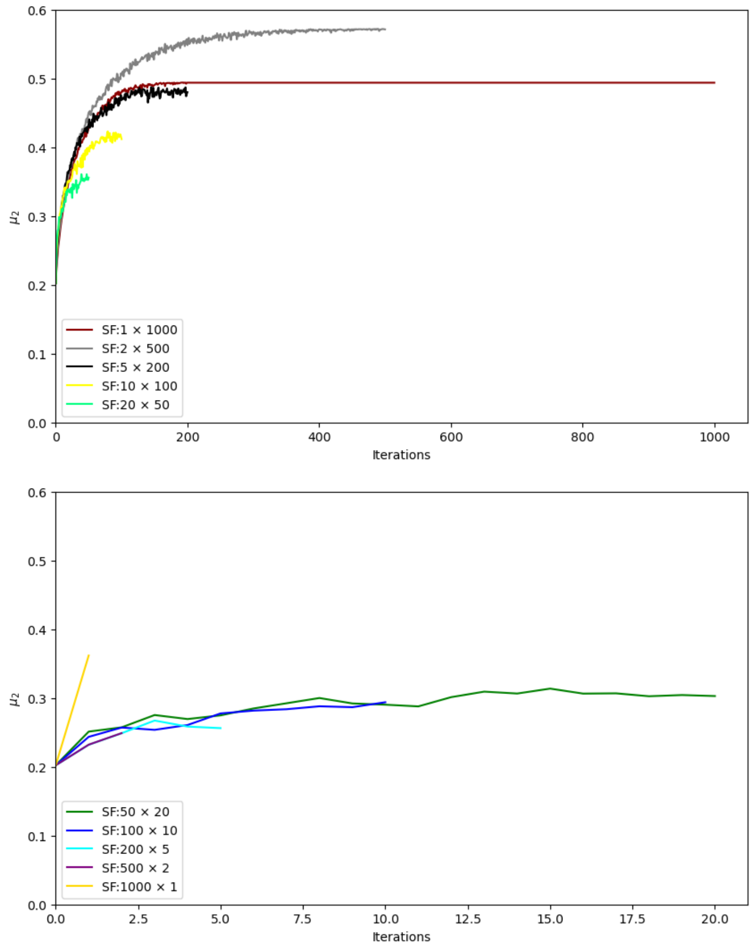

図 4.2: SF ネットワークにおけるリワイヤリング回数に対するラプラシアン行列の第二最小固有値 μ_2 の変化。上図が $l = 1$、$c = 1000$ から $l = 20$、$c = 50$ における μ_2 の変化を示して、下図が $l = 50$、$c = 20$ から $l = 1000$、$c = 1$ のそれぞれに対する μ_2 変化を示す。

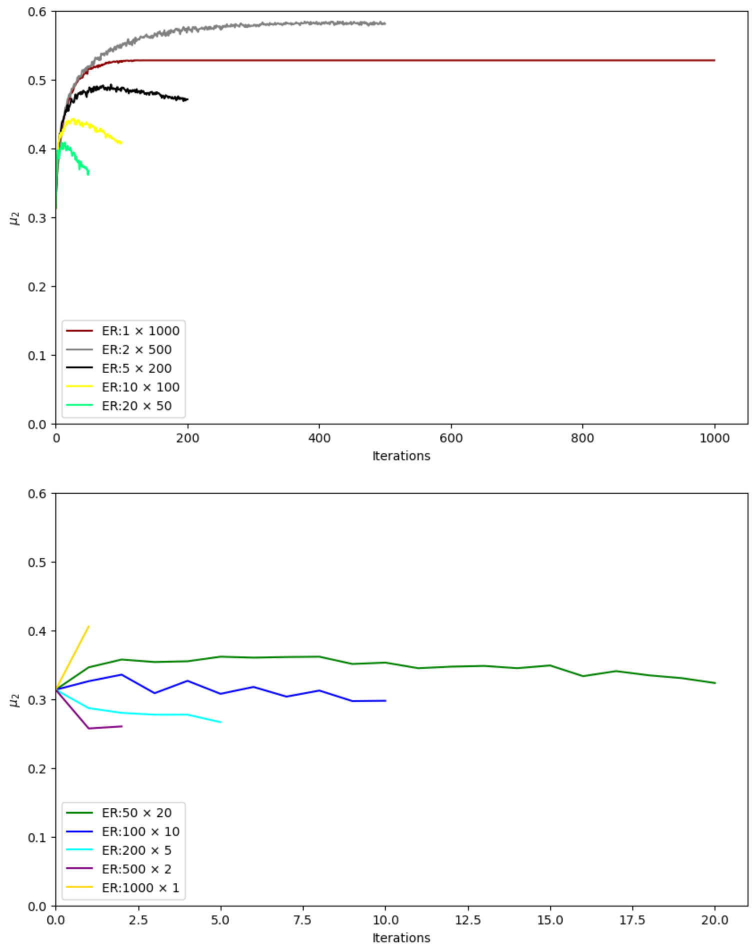

図 4.3: ER ランダムグラフにおけるリワイヤリング回数に対する μ_2 の変化。上図が $l = 1$、$c = 1000$ から $l = 20$、$c = 50$ における μ_2 の変化を示して、下図が $l = 50$、$c = 20$ から $l = 1000$、$c = 1$ のそれぞれに対する μ_2 変化を示す。

4.2 μ_2 とネットワーク結合の頑健性の相関性

図 4.4 に、3 つの戦略に基づく各リワイヤリング後の頑健性指標 R 値の分布を示す。最も R 値が高いのは、SF：rewire（ピンク）と SF：config（紫）の $l = 5$ で $c = 200$ の中間戦略、ER：rewire（水色）と ER：config（緑）の $l = 2$ で $c = 500$ の中間戦略となっている。一方、SF：rewire と SF：config 間における R 値の差（図 4.4 のピンクと紫の差）と、ER：rewire と ER：config 間における R 値の差（図 4.4 の青と緑の差）を見ると、逐次戦略以外において config が rewire より R 値は高いが、その差はあまり大きくない。

また、図 4.5 に、R 値と μ_2 値の傾向を可視化した図を載せる。黒の点線が示すように、R 値と μ_2 値の傾向は類似した傾向を示しているが、例外として一括戦略に基づくリワイヤリング（$l = 1000$、$c = 1$）は $l = 50$、$c = 20$ から $l = 500$、$c = 2$ の中間戦略に基づくリワイヤリング操作よりも μ_2 が大きいにもかかわらず、R 値は全戦略の中で最も小さい。これらの結果から、ネットワーク結合の頑健性向上において、中間戦略が最も R 値を向上させた。但し、図 4.1 では全戦略において config より rewire の方が第二最小固有値 μ_2 が高いにもかかわらず、R 値は config の方が高い傾向にあることから、μ_2 値や次数分布以外の要因が R 値に影響を与えていることが伺える。

次に、μ_2 と R 間の相関関係を図 4.6 に、各凡例の相関係数値を表 4.1 に示す。全ての凡例において、μ_2 の値が大きくなるにつれ、R 値も大きい傾向にある。また、各凡例の相関係数値から μ_2 と R 間には強い正の相関があることが判明した。一方、SF：config（紫）の μ_2 値と R 値の分布の傾向を比較すると、異なる戦略によるリワイヤリングでも μ_2 があまり変化していないにもかかわらず、R 値では戦略毎に変化がみられる。これらの結果をまとめると、μ_2 と R 間は正比例関係ではあるが、μ_2 以外の要因が、R 値に影響を与えていることが伺える。

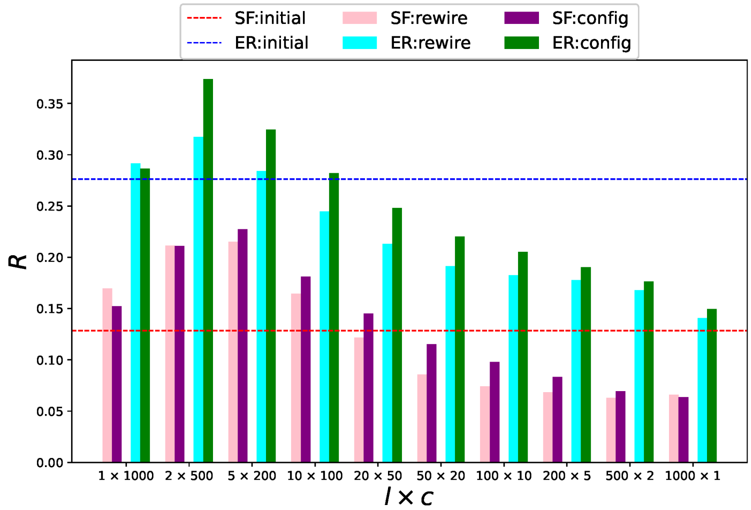

図 4.4: リワイヤリング後の R 値の分布。赤の点線（SF:initial）と青の点線（ER：initial）はそれぞれリワイヤリング前の SF ネットワークと ER ランダムグラフの 100 サンプル平均における μ_2 値を表す。「SF：rewire」と「ER：rewire」はリワイヤリング後のネットワークにおける μ_2 値の 100 サンプル平均、「SF：config」と「ER：config」はリワイヤリング後のネットワークにコンフィギュレーションモデルを適用したネットワークにおける μ_2 値の 100 サンプル平均を表す。

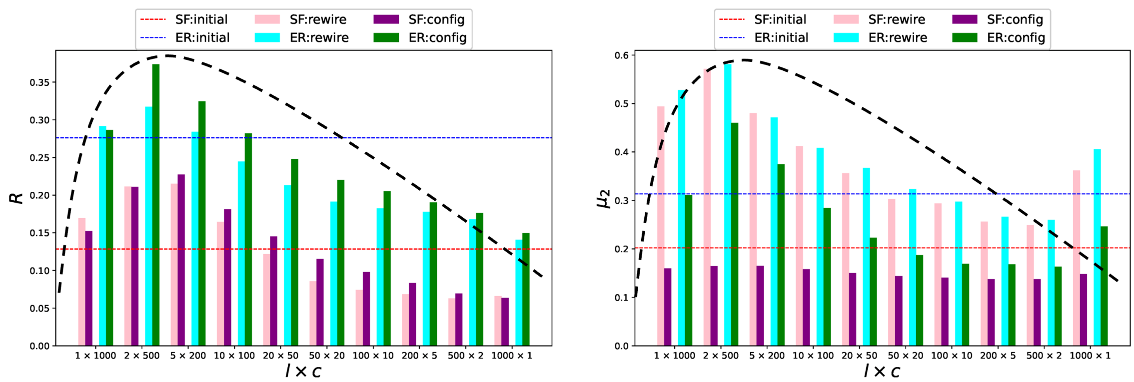

図 4.5: R 値と μ_2 値の傾向を黒の点線で可視化した図

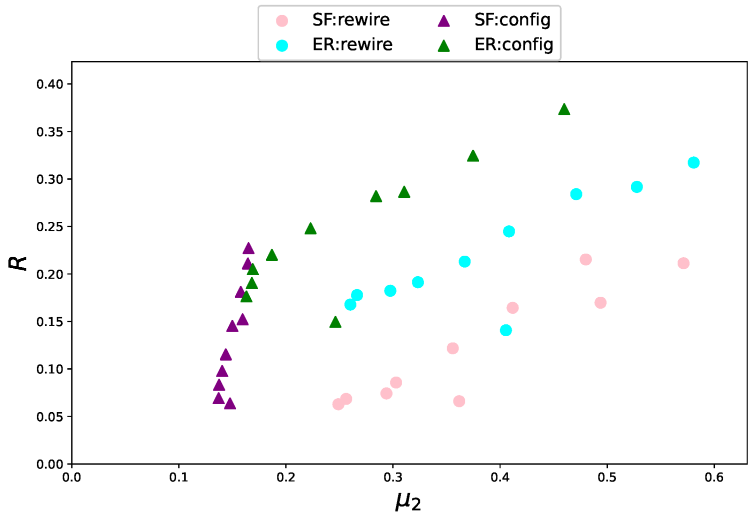

図 4.6: μ_2 と R の相関図

表 4.1: 図 4.4 の相関係数

	rewire	config
SF	0.92	0.91
ER	0.85	0.89

4.3 次数分布 P(k) の変化

図 4.7 と表 4.2 に、リワイヤリング前 (initial) と各戦略に基づくリワイヤリング後の SF ネットワークと ER ランダムグラフにおける次数分布の分散値 σ^2 を示す。また、黒の点線で σ^2 値の変化傾向を示す。具体的な変化の傾向として、SF ネットワークにおける次数分布の σ^2 の変化は、initial から $l=5$、$c=200$ にかけて減少し、$l=5$、$c=200$ から $l=1000$、$c=1$ にかけて増加している。また、ER ランダムグラフにおける次数分布の σ^2 の変化は initial から $l=2$、$c=500$ にかけて減少し、$l=2$、$c=500$ から $l=1000$、$c=1$ にかけて増加している。ここで、図 4.5 の頑健性指標 R 値と第二最小固有値 μ_2 における変化傾向と比較すると、(4.5) の変化の山と図 4.7 の変化の谷が対応関係にあることが視覚的にわかる。これらから、R 値が高いネットワーク程 σ^2 値が小さく、μ_2 を最大化することで σ^2 値が小さくなる傾向がみられる。この結果は、連続変化させた次数分布に従うランダム化したネットワークにおいて、σ^2 値が小さい程 R 値が高くなることを示した研究 [11] を支持するものである。

但し、今回のリワイヤリングの特徴として、一括戦略 ($l=1000$、$c=1$) は比較的高い μ_2 値であるが、全戦略の中で R 値は最低値を、σ^2 は最高値を示している。これは、リワイヤリングにおけるリンク追加するノードペアを選ぶ際、最も第二最小固有ベクトル $\mathbf{u}^{(2)}$ が小さいノードと比較的 $\mathbf{u}^{(2)}$ が大きいノードのペアが選ばれるため、初期ネットワークにおいて $\mathbf{u}^{(2)}$ が最も小さいノードを巨大ハブとした星形構造がリワイヤリングにより生成されていると考えられる。ここで、図 4.8 と図 4.9 に各戦略におけるリワイヤリング後のネットワークの次数分布を示す。μ_2 値が初期ネットワーク (SF:initial,ER:initial) よりも高いネットワークは、低次数が増加し次数分布の幅が狭くなっていることが図 4.8 左と図 4.9 左から視覚的にわかる。逆に、一括戦略を行った後のネットワークの次数分布 (金色の線) をみてみると、他の戦略よりも非常に大きな次数 (k) をもつノードが形成されているため (図 4.8 右と図 4.9 右の赤丸)、σ^2 値は最高値になると考えられる。また、星形構造は、意図的な攻撃で巨大ハブが最優先で除去され、ネットワークが容易に分断されてしまうため、図 4.4 に示すように一括戦略における R 値は最低値になると考えられる。

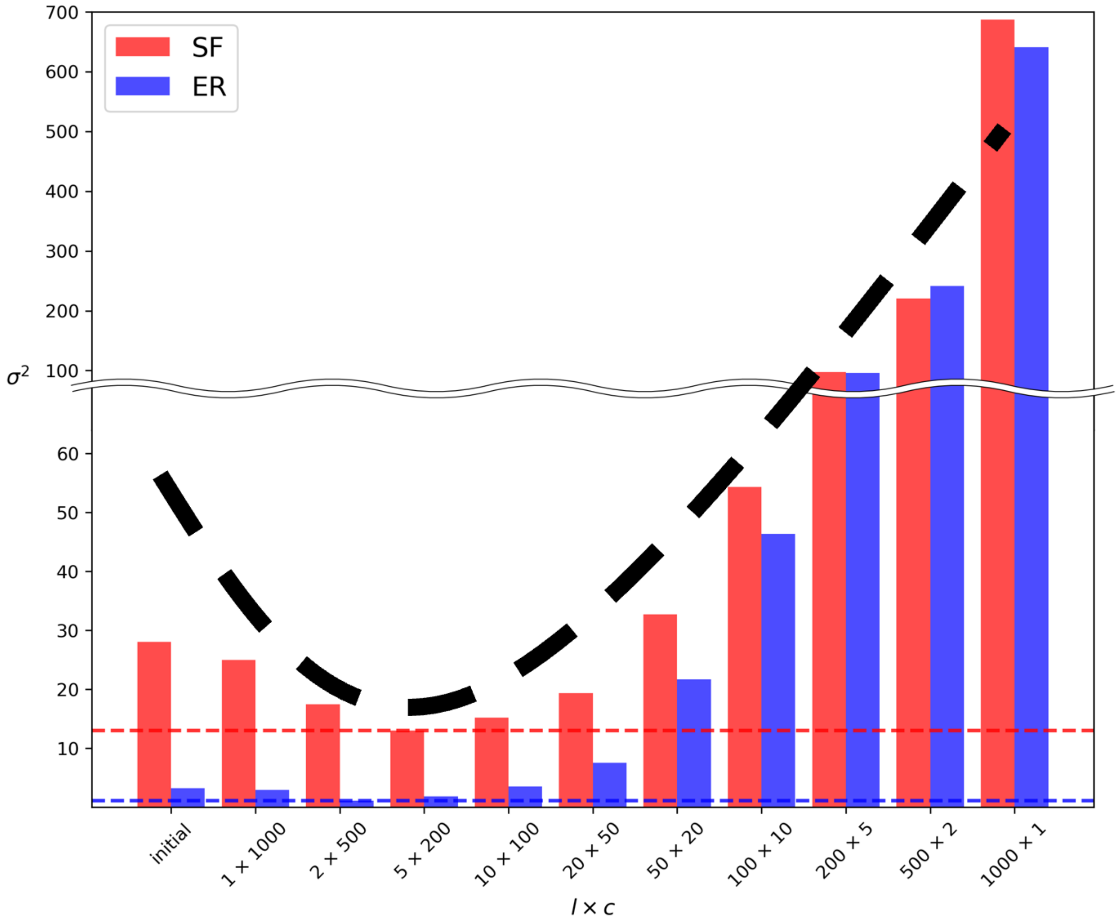

図 4.7: 100 平均に基づく次数分布の分散値 σ^2。黒の点線で値の傾向を、赤の点線と青の点線はそれぞれ SF と ER における σ^2 の最低値を表す。

表 4.2: 図 4.7 の詳細な σ^2 値

$l \times c$	initial	1 × 1000	2 × 500	5 × 200	10 × 100	20 × 50	50 × 20	100 × 10	200 × 5	500 × 2	1000 × 1
SF	27.98	24.98	17.44	13.00	15.15	19.36	32.71	54.29	97.11	220.61	686.88
ER	3.22	2.90	1.06	1.81	3.48	7.50	21.69	46.34	95.78	241.76	640.87

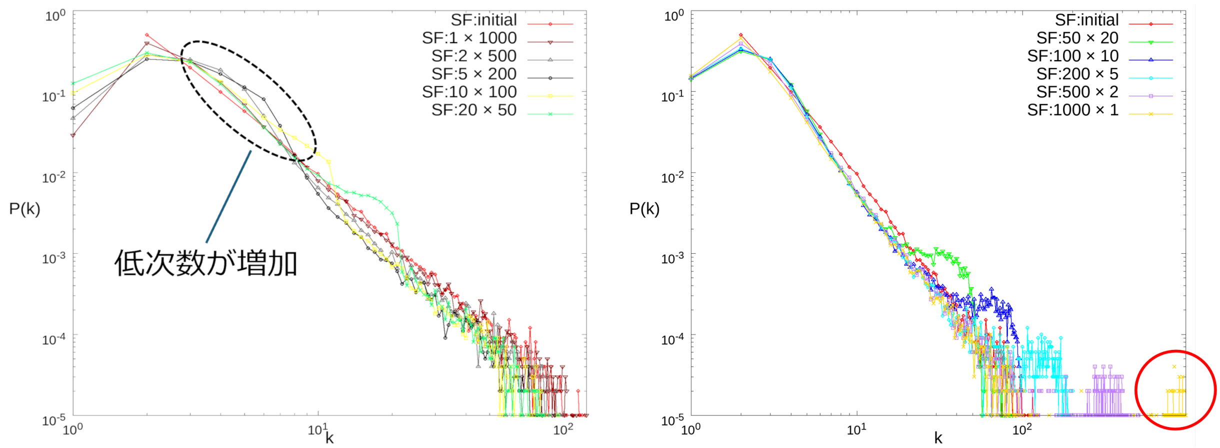

図 4.8: SF ネットワークにおけるリワイヤリング前と全戦略に基づくリワイヤリング後の次数分布。左図がリワイヤリング前（SF：initial）と $l=1$、$c=1000$ から $l=20$、$c=50$ までの、右図が SF：initial と $l=50$、$c=20$ から $l=1000$、$c=1$ までの次数分布に対応している。どちらの図も両対数グラフである。

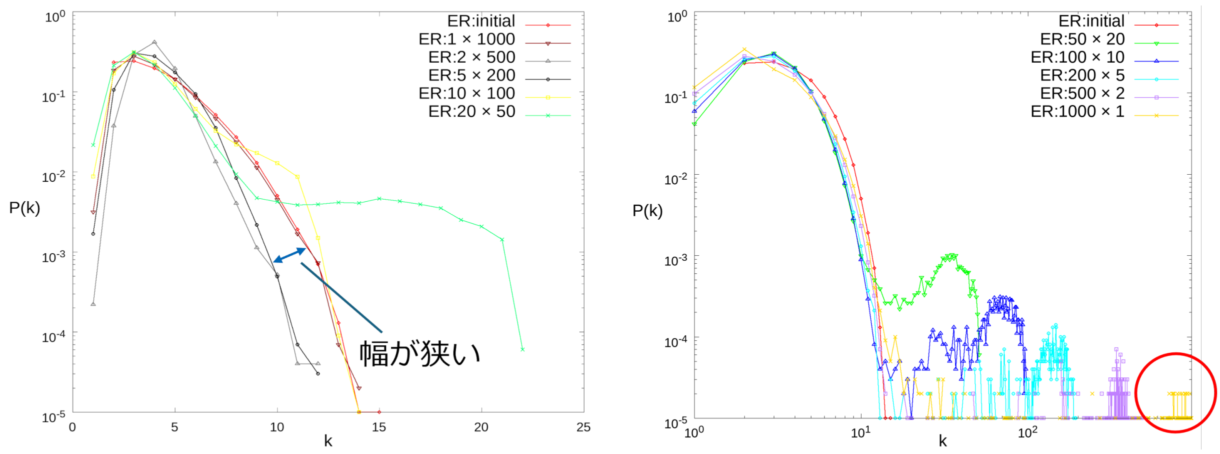

図 4.9: ER ランダムグラフにおけるリワイヤリング前と全戦略に基づくリワイヤリング後の次数分布。左図がリワイヤリング前（ER：initial）と $l=1$、$c=1000$ から $l=20$、$c=50$ までの、右図が ER：initial と $l=50$、$c=20$ から $l=1000$、$c=1$ までの次数分布に対応している。但し、左図は片対数グラフ、右図は両対数グラフである。

第5章　おわりに

本論文では、ラプラシアン行列の第二最小固有値 μ_2 を最大化する複数のリワイヤリング戦略を考えて、様々なシミュレーションを行ったところ、以下の結果を得た。

- μ_2 を最大化するリワイヤリングにおいて、本研究で考えた「逐次戦略」、「一括戦略」、「中間戦略」の中で最も μ_2 値を最大にしたのは「中間戦略」に基づくリワイヤリング手法であった。例外的に、ER ランダムグラフにおいては一度にリワイヤリングする本数が多く、反復回数の少ない中間戦略においては μ_2 値が減少し逆効果もみられた。

- 頑健性指標 R 値と次数分布の分散値 σ^2 は完全に値の変化が連動しており、μ_2 を向上させたリワイヤリングによって、σ^2 値の小さい次数分布をもつネットワークに変化する傾向がみられた。このことは、連続変化させた次数分布に従うランダム化したネットワークにおいて、σ^2 値が小さい程 R 値が高い研究結果と整合する。例外的に、一括戦略ではリワイヤリングにより μ_2 は向上するが、巨大ハブをもつ星形構造が生成されてしまうため、σ^2 値は非常に大きくなり、R 値が最も低くなる。

- μ_2 とネットワーク結合の頑健性指標 R 値には強い正相関がみられた。このことは、μ_2 値の向上がネットワークの二分割を防ぐだけでなく、ネットワーク全体の結合強化に寄与することが明らかになった。

今後の課題として、SF ネットワークと ER ランダムグラフだけでなく、網羅的に連続変化させた次数分布をもつネットワークに μ_2 を最大化するリワイヤリングを適用して、さらに詳細なシミュレーションが必要かもしれない。また、媒介中心性攻撃だけでなく、次数の高いノードを優先して除去する次数中心性攻撃といった他の攻撃における頑健性を調査することも今後の課題として考えられる。

謝辞

本研究をまとめるにあたり、指導教官の林幸雄教授からは様々な助言を賜りました。厚く感謝を申し上げます。また、研究室のメンバーにはゼミ等でご支援いただきました。研究を支えていただいたすべての方に厚く御礼を申し上げ、感謝する次第です。

発表論文　口頭発表

1. Ogawa, T., Hayashi, Y. "Enhancing the robustness of connectivity by maximizing the dominant eigenvalue of Laplacian", *The 13th International Conference on Complex Networks and their Applications*, Book of Abstract（掲載予定）, no. 94, Istanbul, Turkey, Dec. 2024.
2. 小川泰司、林幸雄, " ネットワーク結合耐性を強化するラプラシアン行列の第二最小固有値最大化 ", 第 20 回ネットワーク生態学シンポジウム, 2025 年 3 月 (発表予定).
3. 小川泰司、林幸雄, " ラプラシアン行列の第二固有値最大化によるネットワーク結合の頑健性向上 ", 2025 年電子情報通信学会総合大会, 2025 年 3 月 (発表予定).

参考文献

[1] Kunegis, J. "KONECT: The Koblenz Network Collection." *In Proceedings of the 22nd International Conference on World Wide Web*, WWW '13 Companion, *Association for Computing Machinery*, pp. 1343–1350, NY, USA, 2013.

[2] Barabási, A.-L., and Pósfai, M. *Network Science*. Cambridge University Press, Cambridge, 2016.

[3] Opsahl, T., Agneessens, F., and Skvoretz, J. "Node Centrality in Weighted Networks: Generalizing Degree and Shortest Paths." *Social Networks*, 32(3): pp. 245–251, 2010.

[4] Barabási, A.-L., and Albert, R. "Emergence of Scaling in Random Networks." *Science*, 286, pp. 509–512, 1999.

[5] Amaral, L.A., Scala, A., Barthélémy, M., and Stanley, H.E. "Classes of Behavior of Small-World Networks." *Proceedings of the National Academy of Sciences of the USA*, 97, p. 11149, 2000.

[6] 林幸雄. "自己組織化する複雑ネットワーク-空間上の次世代ネットワークデザイン-." 近代科学社, 2014.

[7] Barabási, A.-L., Albert, R., and Jeong, H. "Mean-field theory for scale-free random networks." *Physical A*, 272, pp. 173–187, 1999.

[8] Albert, R., Jeong, H., and Barabási, A.-L. "Error and attack tolerance of complex networks." *Nature*, 406(6794): pp. 378–382, 2000.

[9] Cohen, R., Erez, K., Ben-Avraham, D., and Havlin, S. "Breakdown of the internet under intentional attack." *Physical Review Letters*, 86(16): pp. 3682, 2001.

[10] Erdős, P., and Rényi, A. "On random graphs. I." *Publicationes Mathematicae Debrecen*, 6, pp. 290–297, 1959.

[11] Chujyo, M., and Hayashi, Y. "Optimal network robustness in continuously changing degree distributions." *Complex Networks and Their Applications XI*,

Proceedings of The Eleventh International Conference on Complex Networks and Their Applications: COMPLEX NETWORKS 2022, 2, 2023.

[12] Newman, M. E. J. "Assortative mixing in networks." *Physical Review Letters*, 89(20): pp. 208701, 2002.

[13] Schneider, C. M., Moreira, A. A., Andrade, J. S., Havlin, S., and Herrmann, H. J. "Mitigation of malicious attacks on networks." *Proceedings of the National Academy of Sciences*, 108(10): pp. 3838–3841, 2011.

[14] Tanizawa, T., Havlin, S., and Stanley, H. E. "Robustness of onion-like correlated networks against targeted attacks." *Physical Review E*, 85:046109, Apr 2012.

[15] Hayashi, Y. "Growing Self-organized Design of Efficient and Robust Complex Networks." *2014 IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, SASO, pp. 50–59, 2014.

[16] Hayashi, Y. "Spatially self-organized resilient networks by a distributed cooperative mechanism." *Physica A: Statistical Mechanics and its Applications*, 457, pp. 255–269, 2016.

[17] Hayashi, Y. "A new design principle of robust onion-like networks self-organized in growth." *Network Science*, 6(1): pp. 54–70, 2018.

[18] Hayashi, Y., and Uchiyama, N. "Onion-like networks are both robust and resilient." *Scientific Reports*, 8:11241, pp. 1–13, 2018.

[19] Braunstein, A., Dall ' Asta, L., Semerjian, G., and Zdeborová, L. "Network dismantling." *Proceedings of the National Academy of Sciences*, 113(44): pp. 12368–12373, 2016.

[20] Chujyo, M., and Hayashi, Y. "A loop enhancement strategy for network robustness." *Applied Network Science*, 6, 3, 2021.

[21] Spielman, D. A. "Spectral graph theory." Chapter 18 of *Combinatorial Scientific Computing* (Eds. U. Naumann and O. Schenk), pp. 495–524, Chapman and Hall/CRC, 2012.

[22] 中山晶一朗、小林俊一、山口裕通. "道路ネットワークの連結性の定量化とその最適補強問題." 土木学会論文集 *D3*, 77-3, 2021.

[23] Krapivsky, P.L., Redner, S., and Leyvraz, F. "Connectivity of Growing Random Networks." *Physical Review Letters*, 85, pp. 4629–4632, 2000.

[24] Liao, F., and Hayashi, Y. "Emergence of robust and efficient networks in a family of attachment models." *arXiv preprint*, arXiv:2110.03176, 2021.

[25] Liao, F., and Hayashi, Y. "Emergence of robust and efficient networks in a family of attachment models." *Physica A: Statistical Mechanics and its Applications*, 599, pp. 127427, 2022.

[26] Callaway, D. S., Hopcroft, J. E., Kleinberg, J. M., Newman, M. E. J., and Strogatz, S. H. "Are randomly grown graphs really random?" *Physical Review E*, 64(4): pp. 041902, Sep 2001.

[27] Catanzaro, M., Boguñá, M., and Pastor-Satorras, R. "Generation of uncorrelated random scale-free networks." *Physical Review E*, 71(2): pp. 027103, Feb 2005.

[28] Fiedler, M. "Algebraic connectivity of graphs." *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

[29] 光澤 駿治, 中山 晶一朗, 小林 俊一, 山口 裕通. "行列木定理を用いた道路ネットワーク評価方法の検討." 土木計画学研究・論文集, 第 38 巻（特集）, 2021.

[30] Mosk-Aoyama, D. "Maximum algebraic connectivity augmentation is NP-hard." *Operations Research Letters*, 36, pp. 677–679, 2008.

[31] Cook, S. "The Complexity of Theorem-Proving Procedures." *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, Shaker Heights, OH, pp. 151–158, 3–5 May 1971. https://doi.org/10.1145/800157.805047

[32] Wang, H., and Mieghem, P.V. "Algebraic connectivity optimization via link addition." *Proceedings of IEEE/ACM Bionetics*, 2008.

[33] Sydney, A., Scoglio, C., and Gruenbacher, D. "Optimizing algebraic connectivity by edge rewiring." *Applied Mathematics and Computation*, 219(10): pp. 5465–5479, 2013.

[34] Freeman, L. C. "A set of measures of centrality based on betweenness." *Sociometry*, pp. 35–41, 1977.

[35] Sun, D.J., Zhao, Y., and Lu, Q.-C. "Vulnerability analysis of urban rail transit networks: A case study of Shanghai, China." *Sustainability*, 7(6), pp. 6919–6936, 2015.

[36] 林幸雄. "複雑ネットワークにおける最適化-超 AI 的な統計物理学アプローチ-." 近代科学社, 2023.

付録A graph-toolの解説

graph-tool は、ブラジル出身のネットワーク科学研究者である Tiago P. Peixoto 博士（Central European University, Hungary）によって開発された、Python 環境で利用可能なネットワーク分析ライブラリである [1]。ネットワーク生成、ノードやリンクの追加・削除、描画といった基本操作に加えて、クラスタリング係数や中心性の測定、連結成分の抽出など、幅広い分析機能を備えている。

graph-tool 以外にも、Python 環境でのネットワーク分析ライブラリとしては NetworkX[2] が有名であり、両者は共にネットワーク解析を目的としている。2 つのライブラリ間で大きく異なる点として、NetworkX が全て Python 言語で記述されているのに対して、graph-tool の計算ルーチンは C++言語で実装されている。一般的に、C++は Python よりも記述が複雑だが、処理速度に優れているため、graph-tool は NetworkX に比べて非常に高速な演算が可能である。実際に、graph-tool の公式サイトに掲載されている graph-tool と他のネットワーク分析ライブラリ間での速度比較 [3]（図 A.1）を行ったところ、graph-tool が圧倒的に高速であることが示されている。

これらの理由から、graph-tool は他の Python におけるネットワーク分析ライブラリと比べ、大規模なネットワーク分析やシミュレーションを効率的に実行できるため、本研究では主に graph-tool を用いてシミュレーションを行った。

Algorithm	graph-tool (16 threads)	graph-tool (1 thread)	igraph	NetworkX
Single-source shortest path	0.0023 s	0.0022 s	0.0092 s	0.25 s
Global clustering	0.011 s	0.025 s	0.027 s	7.94 s
PageRank	0.0052 s	0.022 s	0.072 s	1.54 s
K-core	0.0033 s	0.0036 s	0.0098 s	0.72 s
Minimum spanning tree	0.0073 s	0.0072 s	0.026 s	0.64 s
Betweenness	102 s (~1.7 mins)	331 s (~5.5 mins)	198 s (vertex) + 439 s (edge) (~ 10.6 mins)	10297 s (vertex) 13913 s (edge) (~6.7 hours)

図 A.1: ライブラリ間のパフォーマンス比較。使用環境は Intel(R) Core(TM)i9-9980Hk CPU @ 2.40GHz。[3] から転載。

付録Aに関する参考文献

[1] https://graph-tool.skewed.de/

[2] https://networkx.org/

[3] https://graph-tool.skewed.de/performance

付録B ソースコード

Listing B.1: 必要なライブラリ一覧

```
from graph_tool.all import *
import graph_tool.all as gt
import scipy as sp
import numpy as np
from tqdm import tqdm
import os
```

Listing B.2: ラプラシアン行列の第二最小固有値と第二最小固有ベクトルの算出

```
def no_normalized_laplacian(g):
    L = gt.laplacian(g)
    L_sparse = sp.sparse.csr_matrix(L)
    eigenvalues, eigenvectors = sp.sparse.linalg.eigsh(L_sparse, k
        =2, which='SM', maxiter=100000)
    return eigenvalues[1], eigenvectors[:, 1]
```

Listing B.3: ネットワークの最大連結成分を取得する

```
def max_component(g):
    l = gt.label_largest_component(g)
    u = gt.GraphView(g, vfilt=l)
    sum_u = u.num_vertices()
    return sum_u
```

Listing B.4: 逐次戦略における $\mathbf{u}^{(2)}$ の絶対差が最大のノードペアの取得

```
def max_abs_fiedler(fiedler_vector, g):
    # フィードラーベクトルを次元配列として取得1
    fiedler_vector = np.array(fiedler_vector)

    # 差分行列の計算（|fiedler[i] - fiedler[j）]|
    diff_matrix = np.abs(fiedler_vector[:, None] - fiedler_vector[None, :])

    # 隣接行列を取得
    adjacency = gt.adjacency(g).toarray()

    # エッジがないノードペアのみを対象とするフィルタ
    mask = (adjacency == 0) # 隣接していないノードペアがTrue

    # 差分行列にマスクを適用
    diff_matrix[~mask] = -np.inf # エッジが存在する部分は無効化∞(-)

    # 最大値とそのインデックスを取得
    max_diff = np.max(diff_matrix)
    node_pair = np.unravel_index(np.argmax(diff_matrix), diff_matrix.shape)

    np_tuple = (node_pair)

    # 通常のPython の整数タプルに変換
    python_tuple = tuple(int(x) for x in np_tuple)

    return python_tuple
```

Listing B.5: 逐次戦略における $\mathbf{u}^{(2)}$ の絶対差が最小のノードペアの取得（但し、除去により連結性を失うリンクは除外）

```
def min_abs_fiedler(fiedler_vector, g, removed_edges):
    # フィードラーベクトルを次元配列として取得 1
    fiedler_vector = np.array(fiedler_vector)

    # グラフのエッジリストを取得（削除されたエッジは除外）
    edges = [(e.source(), e.target()) for e in g.edges() if e not
        in removed_edges]

    # エッジリストからノードペアのインデックスを抽出
    node_pairs = np.array(edges, dtype=int) # データ型を明示的に
       整数型に変換

    # 各エッジに対応するノードペアの差分を計算
    diffs = np.abs(fiedler_vector[node_pairs[:, 0]] -
       fiedler_vector[node_pairs[:, 1]])

    # 最小差分と対応するノードペアを取得
    min_diff_idx = np.argmin(diffs)
    min_diff = diffs[min_diff_idx]
    node_pair = tuple(node_pairs[min_diff_idx])

    np_tuple = (node_pair)

    # 通常のPython の整数タプルに変換
    python_tuple = tuple(int(x) for x in np_tuple)

    return python_tuple
```

Listing B.6: 一括戦略と中間戦略における $\mathbf{u}^{(2)}$ の絶対差が大きい順に k 個のノードペアを取得

```
def top_k_max_abs_fiedler(fiedler_vector, g, k):
    """フィードラーベクトルの差分が大きい順に最大
     k 個のノードペアを取得
    """
    # フィードラーベクトルを次元配列として取得1
    fiedler_vector = np.array(fiedler_vector)

    # 差分行列の計算（|fiedler[i] - fiedler[j]）]|
    diff_matrix = np.abs(fiedler_vector[:, None] - fiedler_vector
        [None, :])

    # 隣接行列を取得
    adjacency = gt.adjacency(g).toarray()

    # エッジがないノードペアのみを対象とするフィルタ
    mask = (adjacency == 0) # 隣接していないノードペアがTrue

    # 差分行列にマスクを適用
    diff_matrix[~mask] = -np.inf # エッジが存在する部分は無効化∞
        (-)

    # 上三角行列部分のみを対象にする（重複排除）
    triu_indices = np.triu_indices_from(diff_matrix, k=1)
    valid_diff_values = diff_matrix[triu_indices]

    # 大きい順にk 個のインデックスを取得
    top_k_indices = np.argpartition(valid_diff_values, -k)[-k:]
    top_k_indices = top_k_indices[np.argsort(valid_diff_values[
        top_k_indices])[::-1]]

    # インデックスを次元に戻し、ノードペアに変換2
    node_pairs = [(triu_indices[0][idx], triu_indices[1][idx])
        for idx in top_k_indices]

    return node_pairs
```

Listing B.7: 一括戦略と中間戦略における $\mathbf{u}^{(2)}$ の絶対差が小さい順に k 個のノードペアを取得

```
def top_k_min_abs_fiedler(fiedler_vector, g, k):
    """フィードラーベクトルの差分が小さい順に最小
     k 個のノードペアを取得
    """
    # フィードラーベクトルを次元配列として取得 1
    fiedler_vector = np.array(fiedler_vector)

    # グラフのエッジリストを取得（削除されたエッジは除外）
    edges = [(e.source(), e.target()) for e in g.edges()]

    # ノードペアの順序を固定（小さいノード番号が先に来るようにソート）
    edges = [tuple(sorted(edge)) for edge in edges]

    # 重複を排除
    unique_edges = list(set(edges))

    # 各エッジに対応するノードペアの差分を計算
    node_pairs = np.array(unique_edges, dtype=int) # データ型を明示的に整数型に変換
    diffs = np.abs(fiedler_vector[node_pairs[:, 0]] - fiedler_vector[node_pairs[:, 1]])

    # 小さい順にk 個のインデックスを取得
    top_k_indices = np.argpartition(diffs, k)[:k]
    top_k_indices = top_k_indices[np.argsort(diffs[top_k_indices])]

    # インデックスからノードペアに変換
    top_k_node_pairs = [tuple(node_pairs[idx]) for idx in top_k_indices]

    return top_k_node_pairs
```