Fruchterman–Reingold Layout with Subspace Method as Initial Placement

Hiroki Hamaguchi, Naoki Marumo, Akiko Takeda

***Abstract*—The abstract goes here. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.**

***Index Terms*—Graph drawing, Fruchterman–Reingold algorithm, Random Subspace method.**

I. Introduction

GRAPH is a mathematical structure representing pairwise relationships between objects, and graph drawing is one of the most fundamental tasks in data science. Indeed, Numerous general algorithms have been proposed for graph drawing [1], [2], [3], [4] Among these, one of the most popular strategies is force-directed algorithms.

In force-directed algorithms, the graph is modeled as a physical system of particles. These include methods such as the Kamada–Kawai (KK) layout [5] using shortest path distances for a cost function, and the Fruchterman–Reingold (FR) layout [6], [7], which is the primary focus of this paper.

The FR algorithm is one of the most widely used force-directed algorithms. It is also implemented in many modern graph drawing libraries such as NetworkX [8], Graphviz [9], and igraph [10]. The FR layout is based on a physical model of a system of particles and springs, and FR algorithm seeks the equilibrium of the forces between nodes.

However, since both the KK layout and the FR layout requires to compute the forces or energies between all pairs of vertices, they have a high computational cost of $\mathcal{O}\left(n^2\right)$ per iteration, where n is the number of vertices in the graph. To address this kind of computational burden, several methods have been proposed. One strategy is to approximate the n-body simulation using hierarchical methods such as the fast multipole method [11], the Barnes–Hut approximation [12], multilevel approaches [13], or stress majorization [14].

Another approach is to accelerate the optimization algorithm directly, which aligns with the spirit of our work. Recent researches have accelerated the algorithms for FR layout and KK layout through various methods, including GPU parallel architectures [15], numerical optimization techniques such as L-BFGS [16], and Stochastic Gradient Descent (SGD) [17].

We all are with Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan. A. Takeda is with Center for Advanced Intelligence Project, RIKEN, Tokyo, Japan.

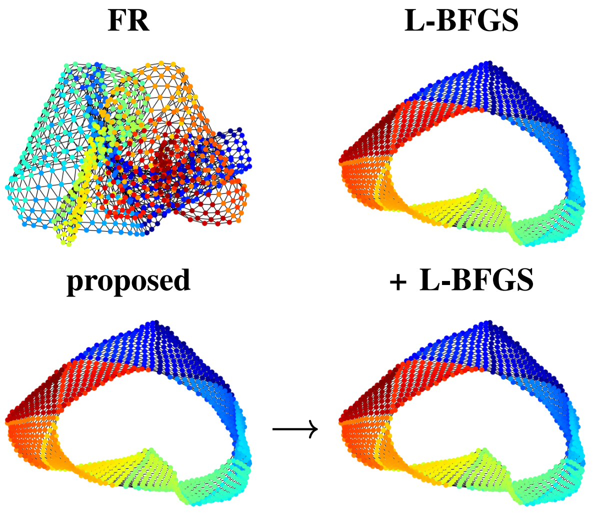

Fig. 1: Comparison of the FR algorithm, L-BFGS, and the proposed method for the `jagmesh1` dataset.

Based on such advances, in this paper, we investigate the ability of another algorithm: subspace methods.

The rest of the paper is organized as follows. In Section II, we define the optimization problem for the FR algorithm. In Section III, we present our research question based on previous works. In Section IV, we propose a new algorithm that utilizes the subspace method for the FR layout. In Section V, we present our experimental results. Finally, we conclude and discuss future work in Section VII.

II. Preliminary

Firstly, in this section, we formulate the FR layout as a continuous optimization problem, and introduce the conventional approaches to this problem, namely the FR algorithm and the L-BFGS method.

A. *Fruchterman–Reingold layout*

Let us define $\mathbb{R}_{>0} := \{x \in \mathbb{R} \mid x > 0\}$, $\mathbb{R}_{\geq 0} := \{x \in \mathbb{R} \mid x \geq 0\}$, and let $W = (w_{i,j}) \in \mathbb{R}^{n\times n}_{\geq 0}$ be an adjacency matrix of a undirected connected graph $G_W = (V, E)$. $V = \{v_i \mid 1 \leq i \leq n\}$ is a set of vertices and $E = \{(v_i, v_j) \mid w_{i,j} > 0\}$ is a set of edges. We call $w_{i,j}$ as a weight of the edge (v_i, v_j).

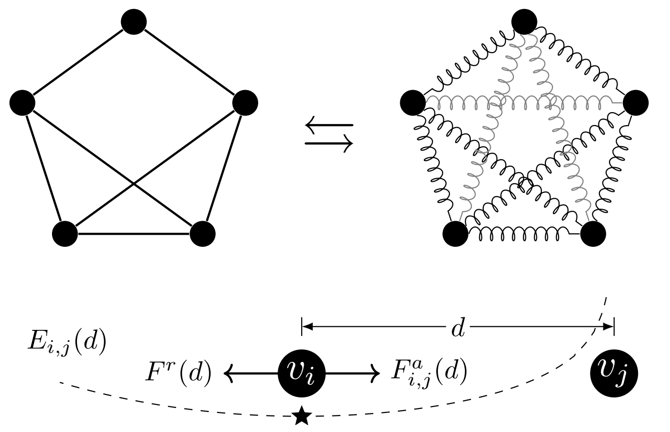

Fig. 2: (Top) Fruchterman–Reingold layout. It models $\mathcal{O}(n^2)$ springs between all pairs of vertices. (Bottom) The equilibrium between attractive force $F^a_{i,j}(d)$ and repulsive force $F^r(d)$ is achieved at $d = k/\sqrt[3]{w_{i,j}}$, which equals k when $w_{i,j} = 1$.

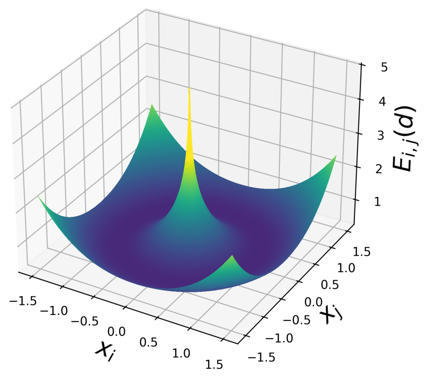

Fig. 3: Energy function $E_{i,j}(d)$ for $x_j = (0, 0)$, $w_{i,j} = 1$ and $k = 1$. Although $E_{i,j}$ is convex as a function of d, it is not convex as a function of x_i. As x_i approaches x_j, the energy function diverges.

The FR algorithm in NetworkX, for example, can handle directed unconnected graphs with $w_{i,j} < 0$, but we will not consider such cases here. For directed graphs, slight modifications of algorithms or converting them to undirected graphs may be effective. For unconnected graphs, algorithms can be applied to each connected component independently. When $w_{i,j} < 0$ for some (i, j), the optimization problem may become unbounded, but with $w_{i,j} \geq 0$ for all (i, j) and the connectivity, the problem is always bounded and solvable.

In summary, the conditions for W is formulated as follows:

$$W \in \mathbb{R}^{n\times n}_{\geq 0}, \quad W = W^\top, \quad G_W \text{ is connected.} \tag{1}$$

Each vertex $v_i \in V$ is assigned to a point on a plane $x_i \in \mathbb{R}^2$ and we define $X = (x_1, \dots, x_n) \in \mathbb{R}^{2\times n}$ as the configuration of the graph. For an parameter k and a distance $d_{i,j} := \|x_i - x_j\|_2$ between two vertices v_i and v_j, Fruchterman and Reingold [6] defined the power of attraction $F^a_{i,j} : \mathbb{R}_{>0} \to \mathbb{R}$ and the power of repulsion $F^r : \mathbb{R}_{>0} \to \mathbb{R}$ as

$$F^a_{i,j}(d) := \frac{w_{i,j} d^2}{k}, \quad F^r(d) := -\frac{k^2}{d}.$$

We refer this force-directed layout as the Fruchterman–Reingold (FR) layout. The energy for these powers $E_{i,j}(d)$ can be defined as

$$\begin{aligned} E_{i,j}(d) &:= \int_0^d F^a_{i,j}(r)\,\mathrm{d}r + \int_\infty^d F^r(r)\,\mathrm{d}r \\ &= \frac{w_{i,j} d^3}{3k} - k^2 \log d. \end{aligned}$$

As a remark, the energy function $E_{i,j}$ is convex as a function of d and minimized when $d^* = k/\sqrt[3]{w_{i,j}}$, but it is not Lipschitz continuous, and is not convex as a function of x_i. Refer to Figure 3.

Now, the optimization problem for FR layout can be formulated as the minimization of the energy function $f : \mathbb{R}^{2\times n} \to \mathbb{R}$, as known as a stress of the graph:

$$\underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} \quad f(X) := \sum_{i<j} E_{i,j}(d_{i,j}) \tag{2}$$

In the following, we will discuss the optimization of this problem.

B. Fruchterman–Reingold algorithm

The Fruchterman–Reingold algorithm [6], the original force-directed algorithm for this layout, can be regarded as a most standard approach to solve the optimization problem (2). As pointed out in [18], the FR algorithm can be regarded as a gradient descent method for the energy function f with a cooling global temperature t.

Let denote $f_i(x) : \mathbb{R}^2 \to \mathbb{R}$ as the energy function for the vertex v_i:

$$f_i(x_i) := \sum_{j\neq i} E_{i,j}(d_{i,j}).$$

The gradient of f_i is

$$\nabla f_i(x_i) = \sum_{j\neq i} \left(\frac{w_{i,j} d_{i,j}}{k} - \frac{k^2}{d^2_{i,j}} \right) (x_i - x_j), \tag{3}$$

which is the sum of forces acting on the vertex v_i.

The pseudo code of the FR algorithm is shown in Algorithm 1. It is based on the original implementation in [6] and implementation in NetworkX [8] with some omitted details.

Algorithm 1: Fruchterman–Reingold algorithm

```
Input: Graph $G_W = (V, E)$
Output: Point configuration $X = (x_1, \dots, x_n)$
define parameters $k, t, dt$, iterations;
$x_i \leftarrow$ random for all $v_i \in V$;
for $j \leftarrow 0$ to iterations do
    compute gradient $\nabla f_i(x_i)$ for all $v_i \in V$;
    $x_i \leftarrow x_i + t\frac{\nabla f_i(x_i)}{\|\nabla f_i(x_i)\|_2}$ for all $v_i \in V$;
    $t \leftarrow t - dt$;
    if convergence condition then
        break;
return pos
```

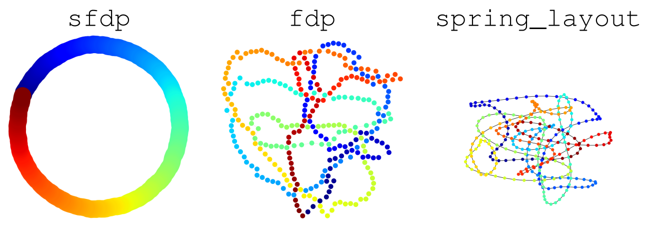

Fig. 4: Comparison of `sfdp` in Graphviz, `fdp` in Graphviz, and `spring_layout` in NetworkX for the circle graph with $|V| = 300$. Every algorithm is run with the default parameters, but both `fdp` and `spring_layout` fails to beautifully visualize the circle.

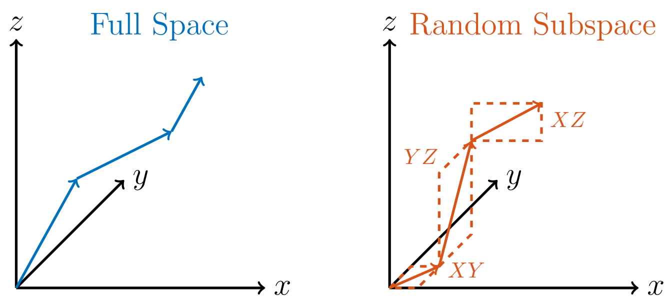

Fig. 5: Visual explanation of Random Subspace Newton TODO

In Algorithm 1, the initial placement of points is determined randomly. In general, under proper normalization of the input, each point is uniformly distributed within a unit square. The parameter t denotes the temperature, which governs the step size of the gradient descent. As the temperature gradually decreases, the algorithm converges to a particular configuration, though this configuration is not necessarily the optimal solution.

C. L-BFGS method

Another approach to solve the optimization problem (2) is to use the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [16]. The L-BFGS method is an optimization technique used for large-scale problems, which approximates the inverse Hessian matrix using only a few recent gradient vectors, making it more memory-efficient than the standard BFGS method [19]. This makes L-BFGS particularly suitable for high-dimensional optimization tasks, and since it is a sophisticated method than the gradient descent method, the superior performance than the FR algorithm is reported in [16].

There are many implementations of L-BFGS available, such as SciPy [20] and C++ L-BFGS [21], [22].

For the optimization problem (2), the L-BFGS method can be applied via flattening the configuration $X \in \mathbb{R}^{2\times n}$ to a vector $\overline{X} \in \mathbb{R}^{2n}$. However, it is worth noting that this method ignores the structure of X, and treats the problem just as a general optimization problem. Thus, we can expect that there could be a room for improvement by leveraging what we have ignored in this L-BFGS method.

III. Research Question

Now, we declare the research question of this paper: "Can we accelerate the optimization process for the FR layout by leveraging the inherent structure of the problem?".

In the previous section, we presented the FR algorithm and L-BFGS method. However, both methods face specific challenges during optimization, leading to inefficiencies and slow convergence as detailed in Section III-A. Addressing these challenges forms a part of our research question, and we aim to introduce a new algorithm capable of overcoming these limitations. To achieve this, we will review key prior studies in Sections III-B and III-C.

A. ねじれの発生

B. SGD for KK layout

Moreover, the effectiveness of Stochastic Gradient Descent (SGD) for Kama–Kawai (KK) layout is well-documented in [17]. The KK layout is a energy based layout, which minimize the energy function f^{KK} defined as

$$f^{\mathrm{KK}}(X) = \sum_{i<j} f^{\mathrm{KK}}_{i,j}(d_{i,j}) = \sum_{i<j} \frac{k}{2}(d_{i,j} - l_{i,j})^2,$$

where k is a constant, and $l_{i,j}$ is the optimal distance between them calculated from the shortest path distance in the graph. The SGD method randomly selects pairs of vertices (i,j) and adjusts their positions to minimize $f^{\mathrm{KK}}_{i,j}$ along the gradient direction, i.e. $x_i \leftarrow x_i - \eta\nabla f^{\mathrm{KK}}_{i,j}(d_{i,j})$ and $x_j \leftarrow x_j + \eta\nabla f^{\mathrm{KK}}_{i,j}(d_{i,j})$ with a learning rate η.

However, in contrast to the KK layout, the FR layout assigns the function $-k^2 \log d_{i,j}$ to all $(i,j) \notin E$, making SGD relatively less effective for the FR layout. Though, the superiority of SGD in the KK layout, which focuses on randomly selected edge, suggests that an optimization method focuses on randomly selected vertex may also be effective in the FR layout. This observation motivates us to explore the application of Random Subspace Newton (RSN) to the FR layout as discussed in the next subsection.

C. Introduction of Random Subspace Newton

Now, we introduce the RSN method, which is not a proposed method itself, but a concept that heavily inspired our proposed algorithm. The RSN method and its variant have been proposed in the context of optimization problems [23], [24], [25], [26].

RSN is a variant of the Newton's method. For a convex function $f(x) : \mathbb{R}^n \to \mathbb{R}$, the Newton's method updates by $x \leftarrow x - \nabla^2 f(x)^{-1}\nabla f(x)$. This requires to compute the inverse of Hessian matrix $\nabla^2 f(x)^{-1} \in \mathbb{R}^{n\times n}$ at each iteration, which poses a high computational cost for large-scale problems. In contrast, RSN focuses on a subspace of dimension s randomly selected from the solution space by a projection matrix P and utilizes the exact Hessian matrix of size $s \times s$ defined on this subspace: $P^\top \nabla^2 f(x) P$. At each iteration, RSN updates the solution by $x \leftarrow x - (P^\top \nabla^2 f(x) P)^{-1} P^\top \nabla f(x)$ if the Hessian matrix is non-singular. Since $s \ll n$, the computational cost per iteration is significantly reduced when we can disregard the cost of selecting the subspace.

The RSN method resembles the stochastic coordinate descent method, which updates only a subset of the variables at each iteration using gradient information. The difference is that RSN uses the Hessian matrix to determine the update direction, bringing the method closer to the Newton method.

Moreover, recent studies have explored its application not only to convex optimization problems but also to non-convex optimization problems [24].

In particular, our problem (2) exhibits a natural affinity with the RSN method, as it inherently defines a subspace of the solution space for the FR layout: $x_i \in \mathbb{R}^2$ for all $v_i \in V$. Although its direct application to the FR layout is not effective as we depicted in Sec. B, we exploit this idea in the proposed algorithm as discussed in the next section.

IV. Proposed Algorithm

以上を基に、本節では我々が提案する Subspace Method による FR layout の最適化手法を述べる。我々の手法は三つの構成要素からなる。Firstly, 我々は最適化問題 (2) を変形し離散最適化問題として hexagonal lattice を用いた簡略化した定式化を行う。これは IV-A 節で述べる。Secondly, 我々は Random に選ばれた頂点についての Newton's direction を用いて、その離散最適化問題を連続緩和によって解く。これは IV-B 節で述べる。Thirdly, 我々はその解を初期解として、最適化問題 (2) を FR algorithm や L-BFGS method で解く。これらの全体を IV-C 節で示す。

A. reduction to the discrete optimization problem

まず、最適化問題 (2) を離散最適化問題に変形する。具体的には、以下のような考察に基づいて、最適化問題を簡略化する。

$$f(X) = \sum_{i<j} f_{i,j}(d_{i,j}) = \sum_{(i,j)\in E} f^a_{i,j}(d_{i,j}) + \sum_{i<j} f^r_{i,j}(d_{i,j})$$

$$\to \text{minimize} \quad \sum_{(i,j)\in E} f^a_{i,j}(d_{i,j}) \quad \text{subject to} \quad f^r_{i,j}(d_{i,j}) \le \epsilon, \quad i<j$$

$$\to \text{minimize} \quad \sum_{(i,j)\in E} f^a_{i,j}(d_{i,j}) \quad \text{subject to} \quad \text{each } x_i \text{ has an exclusive } \epsilon'\text{-ball}$$

Without constraints, $X = 0$ is the optimal solution. closest packing. With hexagonal close-packed structure, f is the sum of $|V|^2 \to |E|$ terms

This idea is partially overlapped with [4], [27], which uses Simulated Annealing (SA) for a circular initial placement.

このような変形を行うのには、主に以下の二つの理由がある。

一つは、主に疎なグラフに対する高速化である。工学的に重要なグラフは往々にして疎なことが多く、そのようなグラフに対しては、目的関数の値を求めるための計算量は $\mathcal{O}\left(|V|^2\right)$ から $\mathcal{O}(|E|)$ に削減され、高速化が期待されるからである。

もう一つは、log による極端な振る舞いを避けるためである。

B. Newton's direction for discrete optimization

次に、Random に選ばれた頂点についての Newton's direction を用いて、その離散最適化問題を連続緩和によって解く。この Newton's direction を求める部分は、本質的に節 III-C で述べた RSN method と同一である。

Random に選ばれた頂点 v_i に対する目的関数 $f_i(x_i)$ の Hessian matrix は、ならし時間計算量 $\mathcal{O}(|E|/n)$ で計算でき、またその逆行列は 2×2 行列の為、計算量は $\mathcal{O}(1)$ である。

そして、そのようにして求めた Newton's direction を用いて、離散最適化問題を連続緩和によって解くことができる。具体的には、Newton's direction による update 後の頂点の位置を求め、それを離散的に配置されている六方格子上の最近傍点に投影し、その点を新たな頂点の位置とする。また、元の位置から新たな頂点まで順に頂点を動かすことにより、離散最適化問題の制約を満たしながら解くことが出来る。図 **??**にその概要を示したので、参照されたい。

以上により、最適化問題 (2) を解くための良質な初期解を得ることができる。

C. pseudo code

The implementation about hexagonal grid is based on [28].

Algorithm 2: Random Subspace Newton for Fruchterman–Reingold layout

```
Input: Graph $G_W = (V, E)$, subspace dimension $s$
Output: Point configuration $X = (x_1, \dots, x_n)$
define parameters $k, t, dt$, iterations;
$x_i \leftarrow$ random for all $v_i \in V$;
for $j \leftarrow 0$ to iterations do
    compute gradient $\nabla f_i(x_i)$ for all $v_i \in V$;
    $x_i \leftarrow x_i - (P^\top \nabla^2 f_i(x_i) P)^{-1} P^\top \nabla f_i(x_i)$ for all $v_i \in V$;
    $t \leftarrow t - dt$;
    if convergence condition then
        break;
return pos
```

V. Numerical Experiment

We used dataset from [29] and MatrixMarket [30].
https://reference.wolfram.com/language/tutorial/GraphDrawingIntroduction.html

A. 網羅的な実験結果

B. 詳細な実験結果

VI. Combination with Other Techniques

頂点の縮約。sfdp. それによって、更に多変数の問題を解くことができる可能性がある。

VII. Discussion

In this paper, we investigated the effectiveness of the Random Subspace Newton method for the Fruchterman–Reingold algorithm.

A. Application to Other Problems

graph isomorphic problem, graph matching problem, graph embedding problem, etc.

VIII. Acknowledgment

The author would like to express our sincere gratitude to PL Poirion and Andi Han for their insightful discussions, which have greatly inspired and influenced this research.

References

[1] W. T. Tutte, "How to Draw a Graph," *Proceedings of the London Mathematical Society*, vol. s3-13, no. 1, pp. 743–767, 1963. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1112/plms/s3-13.1.743

[2] M. Chrobak and T. H. Payne, "A linear-time algorithm for drawing a planar graph on a grid," *Information Processing Letters*, vol. 54, no. 4, pp. 241–246, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/002001909500020D

[3] K. Sugiyama, S. Tagawa, and M. Toda, "Methods for Visual Understanding of Hierarchical System Structures," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 2, pp. 109–125, 1981. [Online]. Available: https://ieeexplore-ieee-org.utokyo.idm.oclc.org/document/4308636

[4] F. Ghassemi Toosi, N. S. Nikolov, and M. Eaton, "Simulated Annealing as a Pre-Processing Step for Force-Directed Graph Drawing," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '16 Companion. Association for Computing Machinery, 2016, pp. 997–1000. [Online]. Available: https://dl.acm.org/doi/10.1145/2908961.2931660

[5] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Information Processing Letters*, vol. 31, no. 1, pp. 7–15, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0020019089901026

[6] T. M. J. Fruchterman and E. M. Reingold, "Graph drawing by force-directed placement," *Software: Practice and Experience*, vol. 21, no. 11, pp. 1129–1164, 1991. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.4380211102

[7] S. G. Kobourov, "Spring Embedders and Force Directed Graph Drawing Algorithms," 2012. [Online]. Available: http://arxiv.org/abs/1201.3011

[8] A. Hagberg, P. J. Swart, and D. A. Schult, "Exploring network structure, dynamics, and function using NetworkX," 2008. [Online]. Available: https://www.osti.gov/biblio/960616

[9] J. Ellson, E. Gansner, L. Koutsofios *et al.*, "Graphviz— Open Source Graph Drawing Tools," in *Graph Drawing*, P. Mutzel, M. Jünger, and S. Leipert, Eds. Springer, 2002, pp. 483–484.

[10] G. Csardi and T. Nepusz, "The igraph software package for complex network research," *InterJournal*, vol. Complex Systems, p. 1695, 2006. [Online]. Available: https://igraph.org

[11] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *Journal of Computational Physics*, vol. 73, no. 2, pp. 325–348, 1987. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021999187901409

[12] J. Barnes and P. Hut, "A hierarchical O(N log N) force-calculation algorithm," *Nature*, vol. 324, no. 6096, pp. 446–449, 1986. [Online]. Available: https://www.nature.com/articles/324446a0

[13] Y. Hu, "Efficient, high-quality force-directed graph drawing," *The Mathematica journal*, vol. 10, pp. 37–71, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:14599587

[14] E. R. Gansner, Y. Koren, and S. North, "Graph Drawing by Stress Majorization," in *Graph Drawing*, D. Hutchison, T. Kanade, J. Kittler *et al.*, Eds. Springer Berlin Heidelberg, 2005, vol. 3383, pp. 239–250. [Online]. Available: http://link.springer.com/10.1007/978-3-540-31843-9_25

[15] P. Gajdoš, T. Ježowicz, V. Uher, and P. Dohnálek, "A parallel Fruchterman–Reingold algorithm optimized for fast visualization of large graphs and swarms of data," *Swarm and Evolutionary Computation*, vol. 26, pp. 56–63, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2210650215000644

[16] H. Hosobe, "Numerical optimization-based graph drawing revisited," in *2012 IEEE Pacific Visualization Symposium*, 2012, pp. 81–88.

[17] J. X. Zheng, S. Pawar, and D. F. M. Goodman, "Graph drawing by stochastic gradient descent," *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 9, pp. 2738–2748, 2019.

[18] D. Tunkelang, "A numerical optimization approach to general graph drawing," Ph.D. dissertation, Carnegie Mellon University, 1999.

[19] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, 1989. [Online]. Available: https://doi.org/10.1007/BF01589116

[20] P. Virtanen, R. Gommers, T. E. Oliphant *et al.*, "SciPy 1.0: Fundamental algorithms for scientific computing in python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[21] Y. Qiu, "Yixuan/LBFGSpp," 2024. [Online]. Available: https://github.com/yixuan/LBFGSpp

[22] N. Okazaki, "Chokkan/liblbfgs," 2024. [Online]. Available: https://github.com/chokkan/liblbfgs

[23] R. Gower, D. Kovalev, F. Lieder, and P. Richtarik, "RSN: Randomized subspace newton," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer *et al.*, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/bc6dc48b743dc5d013b1abaebd2faed2-Paper.pdf

[24] T. Fuji, P.-L. Poirion, and A. Takeda, "Randomized subspace regularized Newton method for unconstrained non-convex optimization," 2022. [Online]. Available: http://arxiv.org/abs/2209.04170

[25] C. Cartis, J. Fowkes, and Z. Shao, "Randomised subspace methods for non-convex optimization, with applications to nonlinear least-squares," 2022. [Online]. Available: http://arxiv.org/abs/2211.09873

[26] R. Higuchi, P.-L. Poirion, and A. Takeda, "Fast Convergence to Second-Order Stationary Point through Random Subspace Optimization," 2024. [Online]. Available: http://arxiv.org/abs/2406.14337

[27] J. Li, Y. Tao, K. Yuan *et al.*, "Fruchterman–reingold hexagon empowered node deployment in wireless sensor network application," *Sensors*, vol. 22, no. 5179, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/14/5179

[28] A. J. Patel, "Hexagonal Grids," Red Blob Games, Tech. Rep., 2013. [Online]. Available: https://www.redblobgames.com/grids/hexagons/

[29] T. A. Davis and Y. Hu, "The University of Florida sparse matrix collection," *ACM Transactions on Mathematical Software (TOMS)*, vol. 38, no. 1, pp. 1–25, 2011.

[30] R. F. Boisvert, R. Pozo, K. Remington *et al.*, "Matrix Market: A web resource for test matrix collections," in *Quality of Numerical Software: Assessment and Enhancement*, R. F. Boisvert, Ed. Springer US, 1997, pp. 125–137. [Online]. Available: https://doi.org/10.1007/978-1-5041-2940-4_9

Appendix A
Optimal Scaling

When we optimize a placement for FR-layout with an initial placement obtained, for instance through KK-layout, scaling the initial placement at first can often yield better results than directly using the unmodified initial placement. In this section, we address the problem of finding the optimal scaling factor that minimizes the energy function for a given configuration.

A. Optimal Scaling Algorithm

Formulating the optimization through scaling, the task reduces to selecting an appropriate scaling factor $x \in \mathbb{R}_{>0}$ that minimizes the following energy function:

$$\begin{aligned}\phi(x) &:= \left(\sum_{i<j} \frac{w_{ij}(xd_{ij})^3}{3k}\right) - k^2 \sum_{i<j} \log(xd_{ij}) \\ &= x^3 \left(\sum_{i<j} \frac{w_{ij} d_{ij}^3}{3k}\right) - \log(x)(k^2 n(n-1)) \\ &\quad - k^2 \sum_{i<j} \log(d_{ij})\end{aligned}$$

$$\phi'(x) = 3x^2 \left(\sum_{i<j} \frac{w_{ij} d_{ij}^3}{3k} \right) - \frac{k^2 n(n-1)}{x}$$

$$\phi''(x) = 6x \left(\sum_{i<j} \frac{w_{ij} d_{ij}^3}{3k} \right) + \frac{k^2 n(n-1)}{x^2}$$

The function $\phi(x)$ is convex, and we can find the optimal scaling factor x by using Newton's method. This algorithm achieves sufficient convergence within a few iterations, and when we pre-compute the coefficients of $\phi(x)$ with $w_{i,j} > 0$, the time complexity is just $\mathcal{O}(|E|)$.

Appendix B
Challenges of the RSN for the FR Algorithm

In this study, we proposed utilizing the subspace method as an initial placement. Then, a natural question can be arose: Can the subspace method alone achieve "fast" optimization throughout the entire process? Specifically, we want to investigate whether the subspace method can optimize the positions of all vertices efficiently without the constraint of the hexagonal lattice.

To address this, we consider the following algorithm as a natural extension and application of the random subspace methods. Namely, we randomly select a vertex v_i, apply Newton's method to f_i using Eq. 3 and its Hessian:

$$\nabla^2 f_i(x_i) = \sum_{j \neq i} \left(\frac{w_{i,j} d_{i,j}}{k} - \frac{k^2}{d_{i,j}^2} \right) I_d + \sum_{j \neq i} \left(\frac{w_{i,j}}{k d_{i,j}} + \frac{2k^2}{d_{i,j}^4} \right) (x_i - x_j)(x_i - x_j)^\top .$$

Then, we update the position of vertex v_i, and repeat this process until convergence. However, this approach fails to work effectively in practice.

We have reached a tentative conclusion that achieving "fast" optimization using the subspace algorithm alone is unlikely. This section outlines some of the reasons behind this negative outcome.

It is important to note that these challenges do not necessarily imply fundamental limitations of the subspace method. On the contrary, improvements based on these identified issues could potentially enhance the effectiveness of the subspace method.

A. Ignorance of other vertex movements

L-BFGS を始めとする、問題全体のヘッシアンを考慮する手法は、ある頂点の位置を更新する際に、他の頂点の動きも考慮すると言える。ここでは、それがどのような意味を持つか論ずる。

B. Inaccuracy of quadratic approximation

まず、前提として、非凸であり、cubic regularized Newton method をはじめとする正則化の追加が求められる。しかし、そのような正則項

二次近似が著しく悪くなる場合がある。しかし、subspace に限定すると、特に問題が生じやすくなる。その一例が Fig. 6 で示すような状況である。

However, unfortunately, the RSN method is not necessarily effective for the FR layout. As shown in Figure 6, although the energy function f_i with respect to the position x_i of each vertex is convex in the FR layout, the overall energy function f with respect to x_i is not convex.

これに対する解決策として、line search の実施などである。しかし、そもそもの Newton's direction が最適解から大きく外れるという問題は、必ずしも解決できない可能性があることには注意されたい。

Appendix C
Another approach based on the Proposed Method

本論文では六方格子に基づいた Subspace Method を提案したが、基本的に同一のアイデアに基づいた他のアプローチも考えられる。

A. Non-randomized approach

一つは、各頂点毎に最適化する際に、ランダムに頂点を選んで最適化していくのではなく、$|V|$ 点全てを同時に最適化する方法である。

こうすることによって、六方格子という制約をよりラフに扱うことが出来る。

六方格子への射影は、MinCostFlow などで $\mathcal{O}\left(|V|^3\right)$ で厳密解が出せる。ソートによる擬似的な射影で、$\mathcal{O}(|V| \log |V|)$ で近似解が出せる。

B. Non-Newton approach

もう一つは、Newton 法を使わずに、勾配法を使う方法である。

以上のいずれも、基本的に性能は落ちると考えているが、実装が簡略化するなどの利点や、イテレーション毎の計算コストが多少減るという利点があり、検討の余地が残されている。

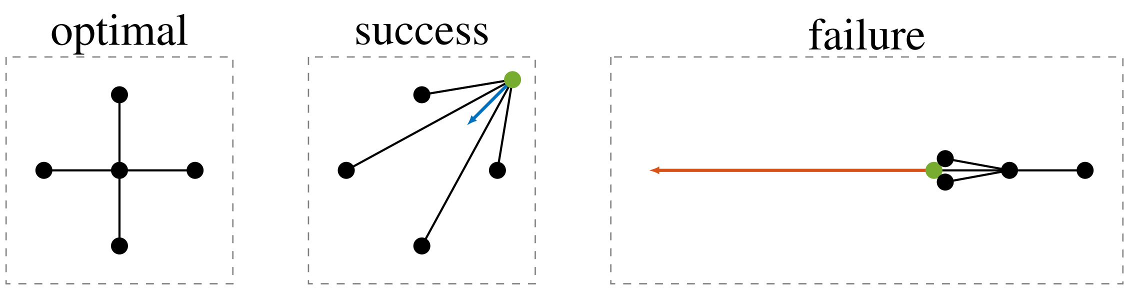

Fig. 6: Visualization of the problem of the subspace method. Let a graph as shown in the left (optimal), where k and all positive edge weights $w_{i,j}$ are set to 1. For the situation in the middle (success), the RSN works effectively. However, in the situation depicted on the right (failure), where the points are set as $x_0 = (0,0), x_1 = (-1,0), x_2 = (-0.85, 0.155), x_3 = (-0.85, -0.155), x_4 = (1,0)$, the Hessian for the subspace of x_1 is approximately $\begin{pmatrix} 1.841 & 0 \\ 0 & 1.159 \end{pmatrix}$. This Hessian, while positive definite and not ill-conditioned, leads to a Newton direction that clearly deviates significantly from the global optimal solution.