



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, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur 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], [?], 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 [7], Graphviz [8], and igraph [9]. 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 $\mathcal{O}(n^2)$ 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 [10], the Barnes–Hut approximation [11], multilevel approaches [12], or stress majorization [13].

Another approach is to accelerate the optimization algorithm directly, which aligns with the spirit of our work. Recent

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.

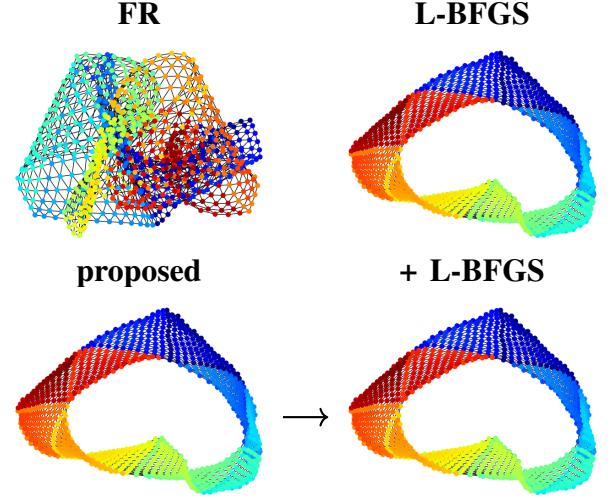


TABLE I: Comparison of the FR algorithm, L-BFGS, and the proposed method for the `jagmesh1` dataset.

researches have accelerated the algorithms for FR layout and KK layout through various methods, including GPU parallel architectures [14], numerical optimization techniques such as L-BFGS [15], and Stochastic Gradient Descent (SGD) [16].

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 algorithm.

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 $W = (w_{i,j}) \in \mathbb{R}_{\geq 0}^{n \times n}$ 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) .

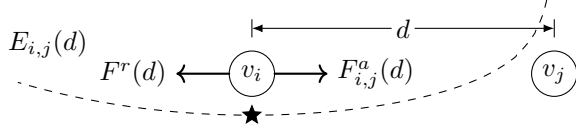


Fig. 1: Fruchterman–Reingold model. The equilibrium of the attractive force $F_{i,j}^a(d)$ and the repulsive force $F^r(d)$ is the optimal distance $d = k / \sqrt[3]{w_{i,j}}$.

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 some weights $w_{i,j}$ are negative, the optimization problem may become unbounded, but with $w_{i,j} > 0$, the problem is always bounded and solvable.

後述の議論の為、 W の条件をまとめると、結局以下のようになる。

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

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 $F_{i,j}^a : \mathbb{R}_{>0} \rightarrow \mathbb{R}$ and the power of repulsion $F^r : \mathbb{R}_{>0} \rightarrow \mathbb{R}$ as

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

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

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

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 2.

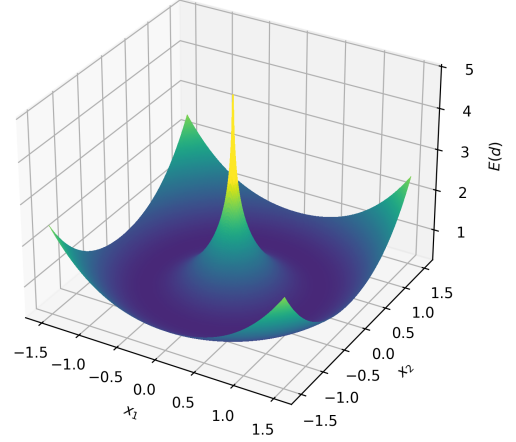


Fig. 2: Energy function $E_{i,j}(d)$ for $w_{i,j} = 1$ and $k = 1$. Although $E_{i,j}$ is convex for d , but not for x_i and x_j .

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

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

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 [17], the FR algorithm can be regarded as a gradient descent method for the energy function f with a cooling global temperature t .

Let denote $f_i(x) : \mathbb{R}^2 \rightarrow \mathbb{R}$ 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

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

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 [7] with some omitted details.

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.



Fig. 3: ねじれ図

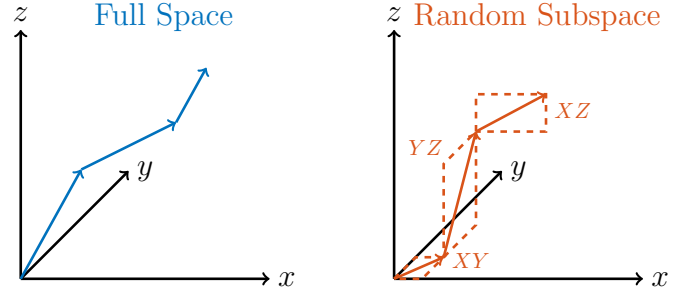


Fig. 4: Visual explanation of Random Subspace Newton

Algorithm 1: Fruchterman–Reingold algorithm

Input: Graph $G_W = (V, E)$
Output: Point configuration $X = (x_1, \dots, x_n)$
 define parameters $k, t, dt, \text{iterations}$;
 $x_i \leftarrow \text{random}$ 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

C. L-BFGS algorithm

Another approach to solve the optimization problem (2) is to use the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm [15]. 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 [?]. 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 [15].

There are many implementations of L-BFGS available, such as SciPy [?] and C++ L-BFGS [18], [19].

For the optimization problem (2), the L-BFGS algorithm can be applied via flattening the configuration X to a vector $\bar{X} \in \mathbb{R}^{2n}$. However, it is worth noting that this method ignores the structure of $X \in \mathbb{R}^{2 \times n}$, 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 formulate the research question of this paper: **“Can we accelerate the optimization process for the FR layout by leveraging the inherent structure of the optimization 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 KK layout is well-documented in [16]. In contrast to the KK layout, where each function is individually determined by $\frac{k_{i,j}}{2}(d_{i,j} - l_{i,j})^2$ based on the actual distance between vertices $d_{i,j}$ and their optimal distance $l_{i,j}$, the FR layout assigns the same value, $-k^2 \log d_{i,j}$, to all pairs of points without direct edges, making SGD relatively less effective for the FR layout.

However, the report in the KK layout, which highlights that optimization focusing on each edge yields favorable results, suggests that optimization focusing on each vertex, such as methods utilizing RSN, may also be effective in the FR layout.

C. Introduction of Random Subspace Newton

Now, we introduce the RSN method. The RSN method and its variant have been proposed in the context of optimization problems [20], [21], [22], [23].

RSN is a variant of the Newton’s method. For a convex function $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, the Newton’s method requires to compute the Hessian matrix $\nabla^2 f(x)$ of size $n \times n$ at each iteration, which poses a high computational cost for large-scale problems. 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 [21].

In this way, the problem exhibits a natural affinity with the RSN method, as it inherently defines a subspace of the solution space for the FR layout. Although its direct application to the FR layout is not effective as we depicted in Sec. B, we exploit this idea 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。

A. reduction to the discrete optimization problem

$$\begin{aligned} f(X) &= \sum_{i < j} f_{i,j}(d_{i,j}) = \sum_{(i,j) \in E} f_{i,j}^a(d_{i,j}) + \sum_{i < j} f_{i,j}^r(d_{i,j}) \\ \rightarrow \text{minimize} \quad & \sum_{(i,j) \in E} f_{i,j}^a(d_{i,j}) \quad \text{subject to} \quad f_{i,j}^r(d_{i,j}) \leq \epsilon, \\ \rightarrow \text{minimize} \quad & \sum_{(i,j) \in E} f_{i,j}^a(d_{i,j}) \quad \text{subject to} \quad \text{each } x_i \text{ has an exclusive } \epsilon' \text{-ball} \end{aligned}$$

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

This idea is partially overlapped with [4], [24], which uses Simulated Annealing (SA) for a circular initial placement.

B. pseudo code

The implementation about hexagonal grid is based on [25].

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 [26] and MatrixMarket [27].

<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 Poiron 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] 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>
- [8] J. Ellson, E. Gansner, L. Koutsofios, S. C. North, and G. Woodhull, “Graphviz—Open Source Graph Drawing Tools,” in *Graph Drawing*, P. Mutzel, M. Jünger, and S. Leipert, Eds. Springer, 2002, pp. 483–484.
- [9] 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>
- [10] 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>
- [11] 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>
- [12] 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>

- [13] E. R. Gansner, Y. Koren, and S. North, “Graph Drawing by Stress Majorization,” in *Graph Drawing*, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and J. Pach, Eds. Springer Berlin Heidelberg, 2005, vol. 3383, pp. 239–250. [Online]. Available: http://link.springer.com/10.1007/978-3-540-31843-9_25
- [14] P. Gajdoš, T. Jeřowicz, 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>
- [15] H. Hosobe, “Numerical optimization-based graph drawing revisited,” in *2012 IEEE Pacific Visualization Symposium*, 2012, pp. 81–88.
- [16] 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.
- [17] D. Tunkelang, “A numerical optimization approach to general graph drawing,” Ph.D. dissertation, Carnegie Mellon University, 1999.
- [18] Y. Qiu, “Yixuan/LBFGSpp,” 2024. [Online]. Available: <https://github.com/yixuan/LBFGSpp>
- [19] N. Okazaki, “Chokkan/liblbfgs,” 2024. [Online]. Available: <https://github.com/chokkan/liblbfgs>
- [20] 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, F. dAlché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/bc6dc48b743dc5d013b1abaebd2faed2-Paper.pdf
- [21] 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>
- [22] 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>
- [23] 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>
- [24] J. Li, Y. Tao, K. Yuan, R. Tang, Z. Hu, W. Yan, and S. Liu, “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>
- [25] A. J. Patel, “Hexagonal Grids,” Red Blob Games, Tech. Rep., 2013. [Online]. Available: <https://www.redblobgames.com/grids/hexagons/>
- [26] 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.
- [27] R. F. Boisvert, R. Pozo, K. Remington, R. F. Barrett, and J. J. Dongarra, “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:

$$\phi(x) := \left(\sum_{i < j} \frac{w_{ij}(xd_{ij})^3}{3k} \right) - k^2 \sum_{i < j} \log(xd_{ij})$$

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

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

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

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 PROBLEM OF RSN FOR FR ALGORITHM

本研究では、Initial Placement として subspace method を用いることを提案したが、自然な疑問として、subspace method 単体で最後まで高速に最適化できないのか、という問いがある。つまり、六方格子という制約を外して、全ての頂点の位置を高速に最適化することができるのか、という問いである。

具体的には、Random Subspace Algorithm の自然な拡張及び適用として、以下のアルゴリズムを考える。則ち、ランダムに頂点 v_i を取り、Eq. 3 と、その Hessian

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

を基に、 f_i に対する Newton 法を適用し、頂点 v_i を更新し、これを繰り返す、というものである。しかし、実はこのままでは上手く動作することはない。

我々は Subspace algorithm 単体で高速な最適化は恐らく不可能であろうという、否定的結論に現在は至っている。その理由の一端を本節では述べる。

なお、これらの問題は必ずしも subspace method の限界を示唆するものではない。寧ろ、これらの問題点に基づいた改良が subspace method の有効性を高める可能性があることには注意されたい。

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. 5 で示すような状況である。

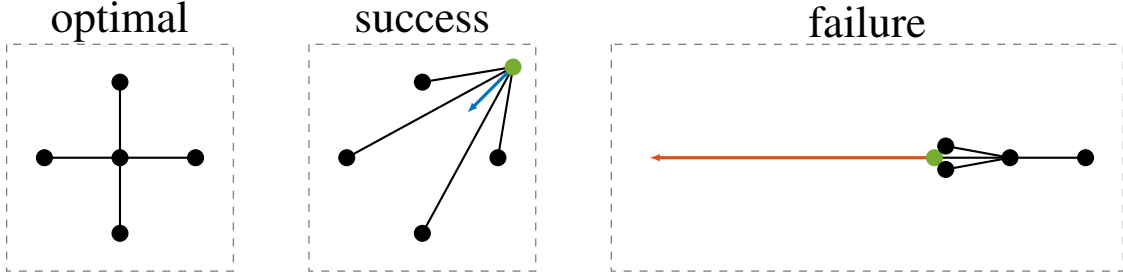


Fig. 5: Visualization of the problem of the subspace method. Let a graph as shown in the left (optimal), where k and all positive edge weights $w_{i,j}$ are set to 1. For the situation in the middle (success), the RSN works effectively. However, in the situation depicted on the right (failure), where 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.

However, unfortunately, the RSN method is not necessarily effective for the FR layout. As shown in Figure 5, 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 などでも $\mathcal{O}(|V|^3)$ 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。