Fruchterman–Reingold Layout with Subspace Method as Initial Placement

Hiroki Hamaguchi Naoki Marumo Akiko Takeda

***Abstract*—If written in Japanese, it indicates that it is in progress. The sentences in orange indicate descriptions that have questionable accuracy or are marked as TODO. The ORCID link is currently broken, so a temporary measure has been taken in the author section. This can be resolved by completely removing the Japanese text and removing dvipdfmx. Be mindful of the differences between cycle, circular, and circle. cycle graph / closest packing of circles. 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, Optimization, Fruchterman–Reingold Layout, Subspace Method, Random Subspace Newton.**

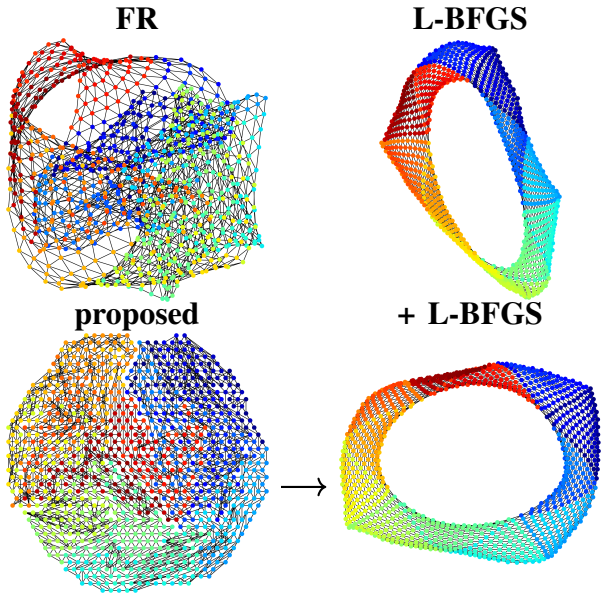

Fig. 1: Comparison of the FR algorithm, L-BFGS, and the proposed method for the `jagmesh1` dataset.

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 kinds of 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, we model a graph as a system of particles with forces acting between them. This class of algorithms includes the Kamada–Kawai (KK) layout [5], Eades' spring embedder [**?**], and the Fruchterman–Reingold (FR) layout [6], [7], which serves as the central focus of this study.

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].

However, both the KK layout and the FR layout suffer from high computational complexity, specifically $\mathcal{O}\left(|V|^2\right)$ per iteration as it is, where $|V|$ denotes the number of vertices. This computational burden makes it difficult to apply these algorithms to large-scale graphs.

To address this kind of burden, several methods have been developed. These include approximating the n-body simulation using multipole expansions [11] or the Barnes–Hut approximation [12], gradually refining the layouts using a multilevel approach [13] and employing stress majorization [14].

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

Based on such advances, in this paper, we investigate the ability of another algorithm: subspace methods. Subspace methods are a class of optimization algorithms that restrict the optimization to a lower-dimensional subspace of the solution space, which can significantly reduce the computational cost per iteration. We propose a new algorithm for the FR layout that leverages the subspace method, and we demonstrate its effectiveness through experiments.

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 VI.

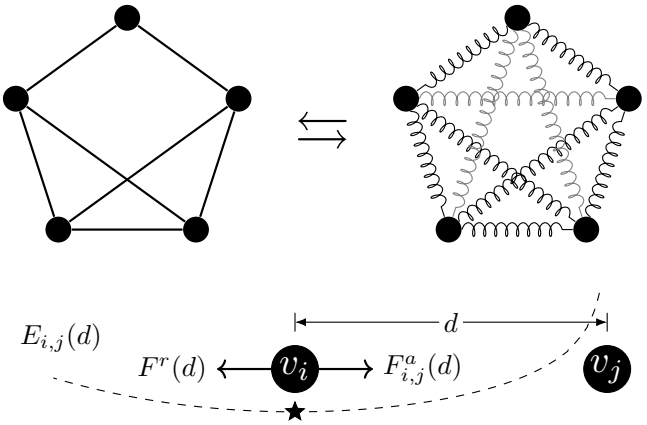

Fig. 2: (Top) Fruchterman–Reingold layout. It models $\mathcal{O}(n^2)$ forces between all pairs of vertices. (Bottom) Forces $F^a_{i,j}(d)$ and $F^r(d)$ works between v_i and v_j,. The equilibrium of them is achieved at $d = k/\sqrt[3]{w_{i,j}}$, which equals k when $w_{i,j} = 1$.

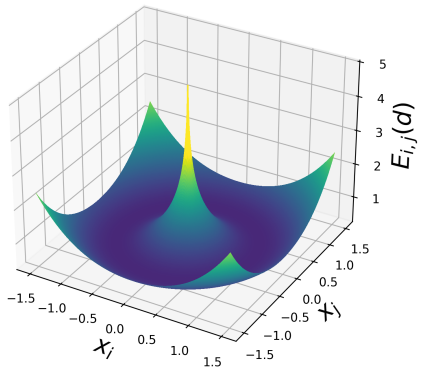

Fig. 3: Energy function $E_{i,j}(d)$ for $x_j = (0, 0)$, $w_{i,j} = 1$ and $k = 1$. Although $E_{i,j}$ is convex as a function of d, it is not convex as a function of x_i. As x_i approaches x_j, the energy function diverges.

II. Preliminary

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 $\mathbb{R}_{>0} \coloneqq \{x \in \mathbb{R} \mid x > 0\}$, $\mathbb{R}_{\geq 0} \coloneqq \{x \in \mathbb{R} \mid x \geq 0\}$, and let $W = (w_{i,j}) \in \mathbb{R}^{n \times n}_{\geq 0}$ be an adjacency matrix of a graph $G_W = (V, E)$, where $V = \{v_i \mid 1 \leq i \leq n\}$ 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).

We will only consider undirected connected graphs with non-negative weights. Although the FR algorithm in NetworkX, for example, can handle directed unconnected graphs with negative weights, this paper does not focus on such cases. 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 negative weights are present, the optimization problem may become unbounded, but with non-negative weights and ensuring the graph's connectivity, the problem is always bounded and solvable. In summary, the conditions for W is formulated as follows:

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

Fruchterman and Reingold [6] proposed a force-directed layout called the Fruchterman–Reingold (FR) layout, as known as a spring layout [8]. Let $x_i \in \mathbb{R}^2$ be the position of the vertex $v_i \in V$, and $X = (x_1, \ldots, x_n) \in \mathbb{R}^{2 \times n}$ be the configuration of the graph. For a parameter k and a distance $d_{i,j} \coloneqq \|x_i - x_j\|_2$ between two vertices v_i and v_j, the attraction force $F^a_{i,j} : \mathbb{R}_{>0} \to \mathbb{R}$ and the repulsion force $F^r : \mathbb{R}_{>0} \to \mathbb{R}$ is defined as

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

FR layout seeks the equilibrium of the forces between all pairs of vertices, as shown in Figure 2.

We can also interrupt forces by its scalar potential [16], in other words, energy $E_{i,j} : \mathbb{R}_{>0} \to \mathbb{R}$, which is defined by

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

As a remark, this 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:

$$\underset{X \in \mathbb{R}^{2 \times n}}{\text{minimize}} \quad f(X) \coloneqq \sum_{i<j} E_{i,j}(d_{i,j}). \tag{3}$$

This is because seeking an equilibrium of the forces is equivalent to minimizing the energy function $E_{i,j}$ for all pairs of vertices. 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 for solving the optimization problem (3). 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:

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

The gradient of f_i is

$$\nabla f_i(x_i) = \sum_{j \neq i} \left(\frac{w_{i,j} d_{i,j}}{k} - \frac{k^2}{d^2_{i,j}} \right)(x_i - x_j), \tag{4}$$

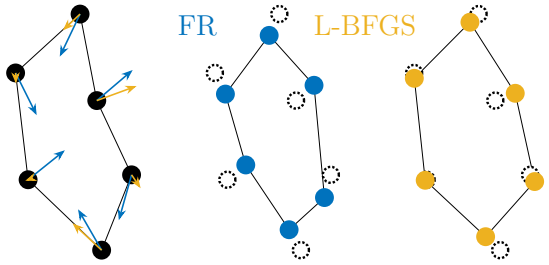

Fig. 4: For the graph on the left, which is in progress of L-BFGS, the FR algorithm and the L-BFGS method are executed for 1 iteration respectively. The L-BFGS achieves a lower $f(X)$, as it approximates the Hessian of f.

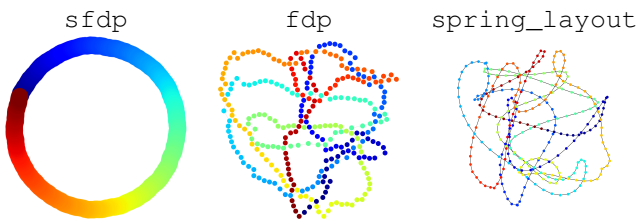

Fig. 5: Comparison of `sfdp` in Graphviz, `fdp` in Graphviz, and `spring_layout` in NetworkX for the cycle graph with $|V| = 300$. Every algorithm is run with the default parameters, but both `fdp` and `spring_layout` fails to beautifully visualize the cycle.

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 $X = (x_1, \ldots, x_n)$
define parameters k, t, dt, iterations;
$x_i \leftarrow$ random position 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 X;

In Algorithm 1, the initial placement of points is determined randomly. Under proper input normalization, each point is uniformly distributed within a unit square in general.

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 (3) 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 (quasi-Newton method) [19]. This makes L-BFGS particularly suitable for high-dimensional optimization tasks. Since it is a more sophisticated method than the gradient descent method, the superior performance of the FR algorithm is reported in [16]. Also refer to Figure 4.

There are many implementations of L-BFGS available, such as SciPy [20] and C++ L-BFGS [21], [22], which we used in our experiments.

For the optimization problem (3), 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 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. twist in the optimization process

First, we observe the challenges appeared in existing methods. We identify that the most significant difficulty in optimizing the FR layout lies in resolving "twists".

The term "twist" does not refer to a mathematically rigorous concept; rather, it describes situations where unnecessary intersections of edges or tangled structures appear in visual representations. Although the usage of the term "twist" is not common, it has been mentioned in works such as [**?**]. Figure 5 illustrates a cycle graph example of this situation.

The presence or absence of "twist" can impact optimization efficiency. In the case shown in Figure 5, even if vertices are disordered, optimization method proceeds relatively smoothly when no intersections between edges exist. However, when "twist" exists, mutual interactions may diminish the gradient, causing the optimization to stagnate in typical continuous optimization processes, making it challenging to resolve the "twist". Therefore, providing an initial placement of points that resolve "twist" as much as possible can significantly influence the efficiency of the optimization process.

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

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

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 $f_{i,j}^{\mathrm{KK}}$ along the gradient direction, i.e. $x_i \leftarrow x_i - \eta\nabla f_{i,j}^{\mathrm{KK}}(d_{i,j})$ and $x_j \leftarrow x_j + \eta\nabla f_{i,j}^{\mathrm{KK}}(d_{i,j})$ 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].

First, we briefly explain the Newton's method. For a convex function $f(x) : \mathbb{R}^n \to \mathbb{R}$, the Newton's method updates x by

$$x \leftarrow x - \nabla^2 f(x)^{-1} \nabla f(x).$$

Since f is convex, the Hessian matrix $\nabla^2 f(x) \in \mathbb{R}^{n\times n}$ is positive semi-definite, and the update direction is a descent direction. Plus, the updated x is a optimal solution of the quadratic approximation of f at x, which ensures the fast convergence of the Newton's method. Newton's method requires to compute the inverse of Hessian matrix $\nabla^2 f(x)^{-1} \in \mathbb{R}^{n\times n}$ at each iteration, posing 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 $P^\top \nabla^2 f(x) P$ 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.

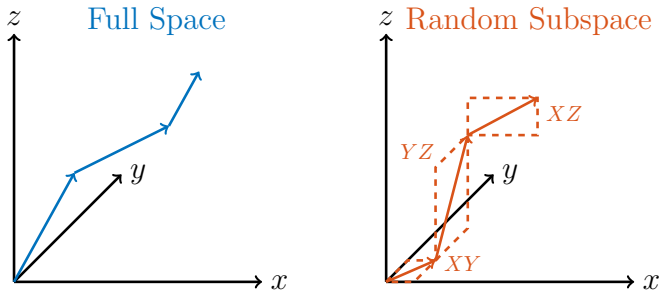

Fig. 6: Visual explanation of Random Subspace Newton TODO

Moreover, recent studies have explored its application not only to convex optimization problems but also to non-convex optimization problems [24], using a regularization term to ensure the convergence of the method.

In particular, our problem (3) exhibits a natural affinity with the RSN method, as it inherently defines a subspace of the solution space X 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. **??**, we exploit this idea in the proposed algorithm as discussed in the next section.

IV. Proposed Algorithm

Based on the research question above, we propose a new algorithm for the FR layout that utilizes the subspace method. Our proposed method is described in three stages. Firstly, in Section IV-A, we reformulate the optimization problem (3) into a simplified discrete optimization problem using a hexagonal lattice. Secondly, in Section IV-B, we present a method to solve the discrete optimization problem through continuous relaxation, using the Newton direction for vertices selected randomly, which is the core of our proposed algorithm. Finally, in Section IV-C, we show the complete framework of the proposed method, where the solution obtained from the previous step is used as the initial solution.

A. reduction to the discrete optimization problem

First, we transform the optimization problem (3) into a constrained continuous optimization problem. When $w_{i,j} = 0$, the energy function $E_{i,j}$ defined in (2) becomes $-k^2 \log d_{i,j}$. Considering the sparsity of many practical graphs ($|E| \ll |V|^2$), simplifying as follow is a reasonable approach:

$$\underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} \quad f(X) = \sum_{(i,j)\in E} \frac{w_{i,j} d_{i,j}^3}{3k} - \sum_{i<j} k^2 \log d_{i,j}. \tag{5}$$

Further, by converting the second term into a constraint, the problem (5) can be approximated such that the objective function can be computed with a complexity dependent on $|E|$ rather than $|V|^2$:

$$\begin{aligned} \underset{X \in \mathbb{R}^{2\times n}}{\text{minimize}} \quad & f^a(X) := \sum_{(i,j)\in E} \frac{w_{i,j} d_{i,j}^3}{3k} \\ \text{subject to} \quad & d_{i,j} \geq \epsilon, \quad \forall (i,j)\,(i<j) \end{aligned} \tag{6}$$

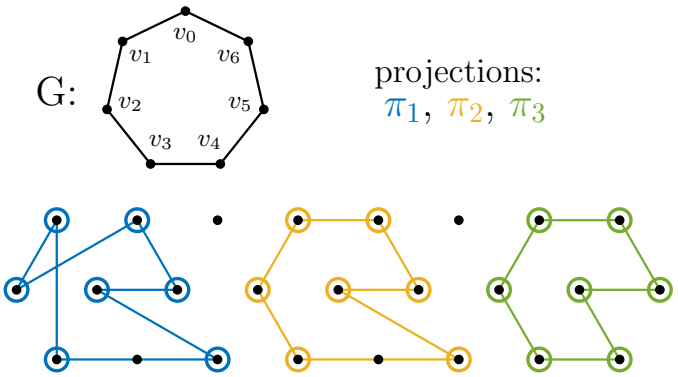

Fig. 7: concept of π

where ϵ is a suitably chosen positive constant. This conversion is reasonable because $-k^2 \log d_{i,j}$ is a convex function such that decreases monotonically with respect to $d_{i,j}$. Thus, for sufficiently large $d_{i,j}$, the value of $-k^2 \log d_{i,j}$ does not grow excessively, ensuring a validity of the approximation.

However, problem (6) still involves $\mathcal{O}\left(|V|^2\right)$ constraints, which negates the advantage of computing the objective function with $\mathcal{O}(|E|)$ complexity. To further simplify, we incorporate the concept of fixed initial configurations for the FR layout [4]. This study reports that a cycle initial placement obtained by Simulated Annealing (SA) allows for a rapid derivation of an initial solution to the optimization problem. Similarly, by simplifying problem (6), we obtain the following discrete optimization problem:

$$\begin{aligned} \underset{\pi : V \to Q}{\text{minimize}} \quad & f^a(X) = \sum_{(i,j)\in E} \frac{w_{i,j} d_{i,j}^3}{3k} \\ \text{subject to} \quad & \|q_i - q_j\|_2 \geq \epsilon, \qquad \forall q_i, q_j \in Q\,(q_i \neq q_j), \\ & x_i = \pi(v_i), \qquad \forall v_i \in V, \\ & \pi(v_i) \neq \pi(v_j), \quad \forall v_i, v_j \in V\,(v_i \neq v_j). \end{aligned} \tag{7}$$

It means that, with a discrete set of points Q such that the points are separated by at least ϵ, we seek a best bijection π from vertices V to Q that minimizes the objective function. By fixing the possible point configuration in advance, the need to check the $\mathcal{O}\left(|V|^2\right)$ constraints is eliminated, reducing the computational complexity to $\mathcal{O}(|E|)$ and thus offering significant speedup. This is exactly why we introduce a discrete optimization problem.

As an example of such a discrete point configuration Q, one could consider n circles of radius ϵ packed in $\mathbb{R}^2$. In this study, however, we adopt a hexagonal lattice. The hexagonal lattice is known for its densest packing structure in space and offers computational simplicity. Although not directly related to this study, prior research [27] has also pointed out the connection between FR layouts and hexagonal lattices. See Figure 7 for reference.

B. the Newton direction for discrete optimization

Next, using the Newton direction for a randomly selected vertex, we solve the discrete optimization problem through continuous relaxation. The computation of the Newton direction in this part is essentially identical to the RSN method described in Section III-C.

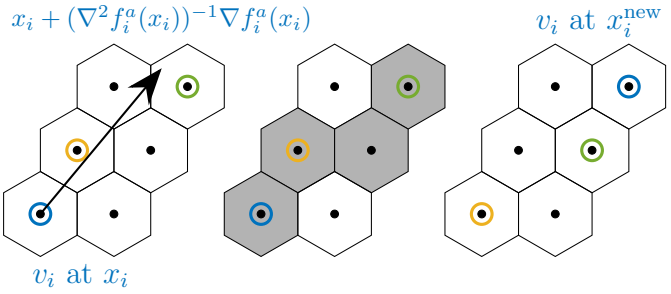

Fig. 8: Visual explanation of the one iteration of the proposed algorithm. Step1. Compute the Newton direction for a randomly selected vertex. Step2. decide the path from the original position to the new position. Step3. move the vertex along the path.

Let the objective function $f_i^a(x_i)$ corresponding to a vertex v_i be

$$f_i^a(x_i) := \sum_{j \neq i} \frac{w_{i,j} \|x_i - x_j\|_2^3}{3k}.$$

Using the Newton direction, we can solve the discrete optimization problem through continuous relaxation, i.e., by updating the position of a vertex v_i as

$$x_i^{\text{new}} \leftarrow x_i - \nabla^2 f_i^a(x_i)^{-1} \nabla f_i^a(x_i),$$

Specifically, the position x_i of the vertex v_i is updated according to the Newton direction, and this new position is projected onto the nearest point on a hexagonal lattice, which is then taken as the new position of the vertex x_i^{new}. Additionally, by sequentially moving vertices from their original positions to the new ones, we can satisfy the constraints of the discrete optimization problem. The overall process is illustrated in Figure 8.

Through this approach, a high-quality initial solution for the optimization problem (3) can be obtained.

C. pseudo code

以上を基に、提案手法の全体像をアルゴリズム 2 に示す。なお、このアルゴリズムの出力は、問題 (3) への解ではなく、離散最適化問題 (7) の解であることに注意されたい。問題 (3) の解は、この出力結果に対し、さらに FR algorithm や L-BFGS method を適用することで得られる。

V. Numerical Experiment

In this section, we evaluate the proposed algorithm by various numerical experiments. 本実験は、Ref. [17] を参考に、次の二つの実験を行う:

1) $|V| = 100$ のランダムグラフに対して、提案手法が有効であることを示す。
2) $|V| = 1000$ のランダムグラフに対して、提案手法が有効であることを示す。

We used dataset from [29] and MatrixMarket [30].

Algorithm 2: Proposed algorithm as as initial placement for the FR layout

Input: Graph $G_W = (V, E)$, subspace dimension s
Output: Point configuration $X = (x_1, \ldots, x_n)$
define parameters $k, Q, \mathit{iterations}$;
set π as a bijection from V to Q randomly;
for $j \leftarrow 0$ **to** *iterations* **do**
  $v_i \leftarrow$ randomly selected vertex from V;
  $q_i \leftarrow \pi(v_i)$;
  $x_i^{\mathrm{new}} \leftarrow x_i - \nabla^2 f_i(x_i)^{-1} \nabla f_i(x_i)$;
  $q_i' \leftarrow$ nearest point of x_i^{new} in Q;
  path $\leftarrow$ path from q_i to q_i';
  update π by moving q_i along the path;
  if *convergence condition* **then**
    break;
$x_i \leftarrow q_i$ for all $v_i \in V$;
return X

A. Experimental Setup

All numerical experiments in this paper were conducted using C++17 compiled by GCC 9.4.0 and a cluster computer powered by Intel(R) Core(TM) i7-10510U CPU with 16 GB RAM. All the codes are available at GitHub [**?**]. The implementation about hexagonal grid is based on [28].

fdp - graphviz version 2.43.0 (0) sfdp - graphviz version 2.43.0 (0)

https://reference.wolfram.com/language/tutorial/GraphDrawingIntroduction.html

B. 網羅的な実験結果

C. 詳細な実験結果

VI. Discussion

In this sections, we discuss the results of the numerical experiments and the potential of the proposed algorithm. Firstly, 我々は提案手法は、$|V| \leq 1000$ に対してのみならず、より大規模な問題に対しても適用可能であると考えている。その際に必要となる他の FR layout の為の手法について、節 VI-A で述べる。Secondly, 提案手法では元問題 (3) 比べ、より複雑かつヒューリスティックな離散最適化問題 (7) を経由したが、その理由について節 VI-B で述べる。Thirdly, 本研究のスコープであるグラフ描画を超えた、別のグラフ最適化問題に対する応用について、節 VI-C で述べる。最後に、we conclude this paper in Section VI-D.

A. Combination with Other Techniques

todo: sfdp のアルゴリズムをきちんと述べる

本論文では、$|V| \leq 1000$ なグラフに対して、提案手法が有効であることを示したが、より大規模な問題に対しても適用可能であると考えられる。例えば Graphviz [**?**] の Scalable Force-Directed Placement (sfdp) では、$|V| > 1000$ などのより大規模なグラフに対し、頂点の縮約を行う事で高速化を図っている。図 5 に示した sfdp の結果も、この手法の有用性を示している。この頂点の縮約という操作は、本提案手法と衝突することなく、両者を組み合わせることが可能であると考えられる。具体的には、縮約後の $|V| < 1000$ となるグラフに対して提案手法を適用することを繰り返せば、より大規模な問題に対しても適用可能であると考えられ、より高速かつより高品質な解を得ることができると考えられる。このような取り組みは、今後の課題の一つとして挙げられる。

B. 何故離散最適化問題を経由するのか

本研究では、離散最適化問題 (7) を経由し、その問題に対する解を subspace method で求め、それを初期解として最適化問題 (3) を解く手法を提案した。ここで、最も自然に湧き上がる疑問としては、何故元の問題 (3) に対して subspace method をそのまま適用せずに、離散最適化問題を経由するのかということが挙げられる。これに対する答えは、Appendix **??** で詳述するが、ここでも簡単に述べる。

我々の考える答えは、本問題に対する subspace method は、大雑把に"twist"を減らすような目的に対して有効であるが、placement 全体を最適化する目的には適していないから、ということである。本研究で用いた subspace method では、一つの頂点のみに着目して、局所的に最善であることは、必ずしも大域的に最適であることを意味しない。故に、subspace method をそのまま適用するだけでは、効率的に全体の最適解を求めることは難しい。

しかし、これは必ずしも subspace method の限界を意味するものではない、と我々は考えている。頂点毎に着目し最適化するというのは自然かつ妥当なアプローチ方法であり、subspace method に更なる改良を加えることで、全体の最適解を効率的に求める可能性があると思われる。

C. Application to Other Problems

グラフを基にした最適化問題は、グラフ描画問題以外にも多く存在する。例えば、pair-wise separable function として分類され、最適化問題の一つとして扱われる問題がある。

今回提案した手法で核となるアイデアは、RSN method の活用であり、頂点毎に最適化を行うことは有用であると実証された。

Originally, this kind of problems are solved by stochastic coordinate descent. Random subspace algorithms might be applicable to these problems

例えば、グラフ描画の問題は、グラフ同型問題との関連性が深い。(Continuous relaxation of) graph isomorphism problem displaying symmetry is at least as difficult as graph isomorphism [**?**] When we draw $G := G_1 \cup G_2$, G displays symmetry if $G_1 \cong G_2$. Graph isomorphism problem with Frank-Wolfe algorithm [**?**]

$$\begin{aligned} \underset{Q}{\text{minimize}} \quad & ||QA - BQ||_F^2 \\ \text{subject to} \quad & Q \in \left\{ Q \in \mathbb{R}^{n \times n} \;\middle|\; Q^\top Q = I_n \right\} \end{aligned}$$

Many-to-Many graph isomorphism [**?**]

$$\begin{aligned} \underset{P}{\text{minimize}} \quad & ||G - PHP^\top||_F^2 \\ \text{subject to} \quad & P \in \left\{ P \in \{0,1\}^{n \times n} \;\middle|\; P1_n = 1_n, P^\top 1_n = 1_n \right\} \end{aligned}$$

D. Conclusion

本研究では、FR layout の最適化問題に対して、subspace method を活用した新たな初期配置手法を提案した。また、その有効性を示すために、数値実験を行い、その結果、提案手法が $|V| \leq 1000$ なグラフに対して有効であることを示した。

提案手法が、FR layout をはじめとするグラフ描画問題、あるいはグラフにまつわる最適化問題に対し、何かしらの示唆を与えることを期待し、本論文を締めくくる。

VII. 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.

networkX の development team に感謝

資金系のサポートについて

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:

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

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

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

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

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 Subspace method 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. 4 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. 9 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 9, 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 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。

Hiroki Hamaguchi Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan.

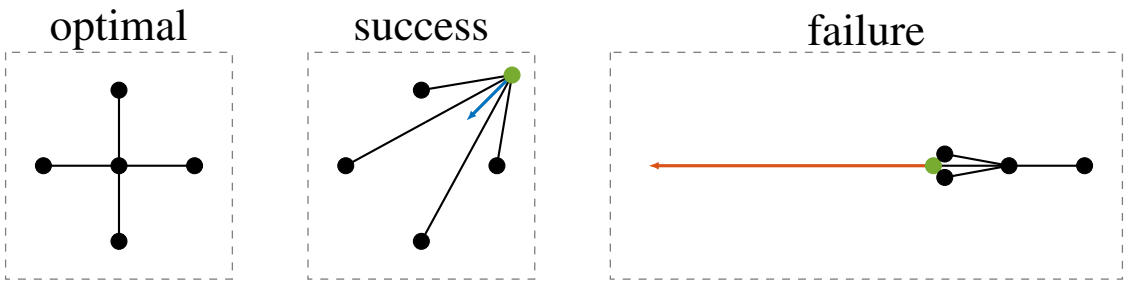

Fig. 9: Visualization of the problem of the subspace method. Let a graph as shown in the left (optimal), where k and all positive edge weights $w_{i,j}$ are set to 1. For the situation in the middle (success), the subspace method works effectively. However, in the situation depicted on the right (failure), 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.

Naoki Marumo Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan.

Akiko Takeda Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan. Center for Advanced Intelligence Project, RIKEN, Tokyo, Japan.