Graph Similarity and Approximate Isomorphism

Martin Grohe
RWTH Aachen University, Aachen, Germany
grohe@informatik.rwth-aachen.de
https://orcid.org/0000-0002-0292-9142

Gaurav Rattan
RWTH Aachen University, Aachen, Germany
rattan@informatik.rwth-aachen.de
https://orcid.org/0000-0002-5095-860X

Gerhard J. Woeginger
RWTH Aachen University, Aachen, Germany
woeginger@informatik.rwth-aachen.de
https://orcid.org/0000-0001-8816-2693

Abstract

The *graph similarity* problem, also known as approximate graph isomorphism or graph matching problem, has been extensively studied in the machine learning community, but has not received much attention in the algorithms community: Given two graphs G, H of the same order n with adjacency matrices A_G, A_H, a well-studied measure of similarity is the *Frobenius distance*

$$\mathrm{dist}(G, H) := \min_{\pi} \|A_G^{\pi} - A_H\|_F,$$

where π ranges over all permutations of the vertex set of G, where A_G^{π} denotes the matrix obtained from A_G by permuting rows and columns according to π, and where $\|M\|_F$ is the Frobenius norm of a matrix M. The (weighted) graph similarity problem, denoted by GSim (WSim), is the problem of computing this distance for two graphs of same order. This problem is closely related to the notoriously hard *quadratic assignment problem* (QAP), which is known to be NP-hard even for severely restricted cases.

It is known that GSim (WSim) is NP-hard; we strengthen this hardness result by showing that the problem remains NP-hard even for the class of trees. Identifying the boundary of tractability for WSim is best done in the framework of linear algebra. We show that WSim is NP-hard as long as one of the matrices has unbounded rank or negative eigenvalues: hence, the realm of tractability is restricted to positive semi-definite matrices of bounded rank. Our main result is a polynomial time algorithm for the special case where the associated (weighted) adjacency graph for *one of the matrices* has a bounded number of twin equivalence classes. The key parameter underlying our algorithm is the clustering number of a graph; this parameter arises in context of the spectral graph drawing machinery.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Graph Similarity, Quadratic Assignment Problem, Approximate Graph Isomorphism

Digital Object Identifier 10.4230/LIPIcs.MFCS.2018.20

Related Version A full version of the paper is available at `https://arxiv.org/abs/1802.08509`.

43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018).
Editors: Igor Potapov, Paul Spirakis, and James Worrell; Article No. 20; pp. 20:1–20:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Graph isomorphism has been a central open problem in algorithmics for the last 50 years. The question of whether graph isomorphism is in polynomial time is still wide open, but at least we know that it is in quasi-polynomial time [4]. On the practical side, the problem is largely viewed as solved; there are excellent tools [9, 15, 21, 22] that efficiently decide isomorphism on all but very contrived graphs [25]. However, for many applications, notably in machine learning, we only need to know whether two graphs are "approximately isomorphic", or more generally, how "similar" they are. The resulting *graph similarity* problem has been extensively studied in the machine learning literature under the name *graph matching* (e.g. [1, 10, 14, 29, 30]), and also in the context of the schema matching problem in database systems (e.g. [23]). Given the practical significance of the problem, surprisingly few theoretical results are known. Before we discuss these known and our new results, let us state the problem formally.

Graph Similarity. It is not obvious how to define the distance between two graphs, but the distance measure that we study here seems to be the most straightforward one, and it certainly is the one that has been studied most. For two n-vertex graphs G and H with adjacency matrices A_G and A_H, we define the *Frobenius distance* between G and H to be

$$\mathrm{dist}(G,H) := \min_{\pi} \|A_G^{\pi} - A_H\|_F. \tag{1}$$

Here π ranges over all permutations of the vertex set of G, A_G^{π} denotes the matrix obtained from A_G by permuting rows and columns according to π, and the norm $\|M\|_F := \sqrt{\sum_{i,j} M_{ij}^2}$ is the *Frobenius norm* of a matrix $M = (M_{ij})$. Note that $\mathrm{dist}(G,H)^2$ counts the number of edge mismatches in an optimal alignment of the two graphs. The *graph similarity problem*, denoted by GSim, is the problem of computing $\mathrm{dist}(G,H)$ for graphs G, H of the same order, or, depending on the context, the decision version of this problem (decide whether $\mathrm{dist}(G,H) \leq d$ for a given d). We can easily extend the definitions to weighted graphs and denote the *weighted graph similarity problem* by WSim. In practice, this is often the more relevant problem. Instead of the adjacency matrices of graphs, we may also use the Laplacian matrices of the graphs to define distances. Recall that the *Laplacian matrix* of a graph G is the matrix $L_G := D_G - A_G$, where D_G is the diagonal matrix in which the entry $(D_G)_{ii}$ is the degree of the ith vertex, or in the weighted case, the sum of the weights of the incident edges. Let $\mathrm{dist}_L(G,H) := \min_{\pi} \|L_G^{\pi} - L_H\|_F$ be the corresponding distance measure. Intuitively, in the definition of $\mathrm{dist}_L(G,H)$ we prefer permutations that map vertices of similar degrees onto one another. Technically, $\mathrm{dist}_L(G,H)$ is interesting, because the Laplacian matrices are positive semidefinite (if the weights are nonnegative). Both the (weighted) similarity problem and its version for the Laplacian matrices are special cases of the problem MSim of computing $\min_P \|A - PBP^{-1}\|_F$ for given symmetric matrices $A, B \in \mathbb{R}^{n\times n}$. In the Laplacian case, these matrices are positive semidefinite.[1]

The QAP. The graph similarity problem is closely related to *quadratic assignment problem* (QAP) [6]: given two $(n \times n)$-matrices A, B, the goal is to find a permutation $\pi \in S_n$ that minimizes $\sum_{i,j} A_{ij} B_{\pi(i)\pi(j)}$. The usual interpretation is that we have n *facilities* that we

[1] Note that the notion of similarity that we use here has nothing to do with the standard notion of "matrix similarity" from linear algebra.

want to assign to n *locations*. The entry A_{ij} is the *flow* from the ith to the jth facility, and the entry B_{ij} is the *distance* from the ith to the jth location. The goal is to find an assignment of facilities to locations that minimizes the total cost, where the cost for each pair of facilities is defined as the flow times the distance between their locations. The QAP has a large number of real-world applications, as for instance hospital planning [11], typewriter keyboard design [27], ranking of archeological data [18], and scheduling parallel production lines [13]. On the theoretical side, the QAP contains well-known optimization problems as special cases, as for instance the Travelling Salesman Problem, the feedback arc set problem, the maximum clique problem, and all kinds of problems centered around graph partitioning, graph embedding, and graph packing.

In the maximization version MAX-QAP of QAP, the objective is to maximize the quantity $\sum_{i,j} A_{ij}B_{\pi(i)\pi(j)}$ (see [19, 24]). Both QAP and MAX-QAP are notoriously hard combinatorial optimization problems, in terms of practical solvability [28] as well as in terms of theoretical hardness results even for very restricted special cases [5, 8, 7]. It is easy to see that MSIM is equivalent to MAX-QAP, because in reductions between QAP and MSIM the sign of one of the two matrices is flipped. Most of the known results for GSIM and its variants are derived from results for (MAX)QAP.

Previous Work. It seems to be folklore knowledge that GSIM is NP-complete. For example, this can be seen by a reduction from the Hamiltonian path problem: take G to be the n-vertex input graph and H a path of length n; then $\mathrm{dist}(G,H) \leq \sqrt{|E(G)| - n}$ if and only if G has a Hamiltonian path. By the same argument, we can actually reduce the subgraph isomorphism problem to GSIM. Arvind, Köbler, Kuhnert, and Vasudev [3] study several versions of what they call *approximate graph isomorphism*; their problem MIN-PGI is the same as our GSIM. They prove various hardness of approximation results. Based on an earlier QAP-approximation algorithm due to Arora, Frieze, and Kaplan [2], they also obtain a quasi-polynomial time approximation algorithm for the related problem MAX-PGI. Further hardness results were obtained by Makarychev, Manokaran, and Sviridenko [19] and O'Donnell, Wright, Wu, and Zhou [26], who prove an average case hardness result for a variant of GSIM problem that they call *robust graph isomorphism*. Keldenich [16] studied the similarity problem for a wide range matrix norms (instead of the Frobenius norm) and proved hardness for essentially all of them.

Our (hardness) results. So where does all this leave us? Well, GSIM is obviously an extremely hard optimization problem. We start our investigations by adding to the body of known hardness results: we prove that GSIM remains NP-hard even if both input graphs are trees (Theorem 8). Note that in strong contrast to this, the subgraph isomorphism problem becomes easy if both input graphs are trees [20]. The reduction from Hamiltonian path sketched above shows that GSIM is also hard if one input graph is a path. We prove that GSIM is tractable in the very restricted case that one of the input graphs is a path and the other one is a tree (Theorem 9).

As WSIM and MSIM are essentially linear algebraic problems, it makes sense to look for algebraic tractability criteria. We explore bounded rank (of the adjacency matrices) as a tractability criteria for WSIM and MSIM. Indeed, the NP-hardness reductions for GSIM involve graphs which have adjacency matrices of high rank (e.g. paths, cycles). We show that the problem GSIM (and WSIM) remains NP-hard as long as one of the matrices has unbounded rank or negative eigenvalues. (Theorems 10, 11 and 12). Consequently, the realm of tractability for WSIM (and MSIM) is restricted to the class of positive semi-definite (PSD) matrices of bounded rank.

Block Partition Structure. We feel that for a problem as hard as QAP or MSIM, identifying any somewhat natural tractable special case is worthwhile. Since the spectral structure of PSD matrices of bounded rank is quite limited, we consider combinatorial restrictions: in particular, restricting the block structure of these matrices is a natural line of investigation.

Given a weighted graph G, we call two vertices *twins* if they have identical (weighted) adjacency to every vertex of G. The *twin-equivalence partition* of $V(G)$, corresponding to this equivalence relation, induces a block structure on the adjacency matrix A_G. Indeed, if $S_1 \dot\cup \ldots \dot\cup S_p = V(G)$ are the twin-equivalence classes, the submatrix $A_G[S_i, S_j]$ is a constant matrix. Hence, the rows and columns of the matrix A_G can be simultaneously rearranged to yield a $p \times p$ block matrix. The number of twin-equivalence classes will be an important parameter of our interest: we denote this parameter by $\tau(G)$.

Our (algorithmic) results. Our main result is a polynomial time algorithm for MSIM if both input matrices are positive semidefinite and have bounded-rank, and where *one of the input matrices* has a bounded number of twin-equivalence classes. Formally, we prove the following theorem. Here, the $\widetilde{\mathcal{O}}$ notation hides factors polynomial in the input representation.

▶ **Theorem 1.** *The problem* MSIM *can be solved in* $\widetilde{\mathcal{O}}(n^{kp^2})$ *time where*

(i) *the input matrices are* $n \times n$ *PSD matrices of rank at most* k*, and*

(ii) *one of the input matrices has at most* p *twin-equivalence classes.*

For the proof of Theorem 1, we can re-write the (squared) objective function as $\|AP - PB\|_F^2$, where P ranges over all permutation matrices. This is a convex function, and it would be feasible to minimize it over a convex domain. The real difficulty of the problem lies in the fact that we are optimizing over the complicated discrete space of permutation matrices. Our approach relies on a linearization of the solution space, and the key insight (Lemma 19) is that the optimal solution is essentially determined by polynomially many hyperplanes. To prove this, we exploit the convexity of the objective function in a peculiar way.

2 Preliminaries

We denote the set $\{1, \ldots, n\}$ by $[n]$. Unless specified otherwise, we will always assume that the vertex set of an n-vertex graph G is $[n]$. We denote the degree of a vertex v by $d_G(v)$.

Twins. Given a $n \times n$ symmetric matrix A with real entries, let G_A denote the associated weighted adjacency graph. Two vertices are called *twins* if they have identical (weighted) adjacency to every vertex in the graph. Hence, two vertices labeled $i, j \in [n]$ are twins if and only if $A_{il} = A_{jl}$ for all $l \in [n]$. This is an equivalence relation; call the resulting partition of the vertex set as the *twin-equivalence partition*. The number of twin-equivalence classes will be an important parameter of our interest: we denote this parameter by $\tau(G)$. In these definitions, we use the matrix A and its adjacency graph A_G interchangeably. This allows us to define $\tau(A)$ for a matrix A to be $\tau(G_A)$, for the associated weighted adjacency graph G_A. The connection with block structure of the matrix is straighforward: observe that we can simultaneously rearrange the rows and columns of A to obtain a $\tau(A) \times \tau(A)$ block matrix W. If $S_1 \dot\cup \ldots \dot\cup S_p = [n]$ be the twin-equivalence partition, the block W_{lm} (where $l, m \in [\tau(A)]$) is the adjacency matrix for the induced subgraph $G_A[S_l, S_m]$. Moreover, the definition of twin-equivalence partition implies that this subgraph is a weighted complete bipartite graph.

Matrices. Given an $m \times n$ matrix M, the i^{th} row (column) of M is denoted by M^i (M_i). The multiset $\{M^1, \ldots, M^m\}$ is denoted by $\mathrm{rows}(M)$. Given $S \subseteq [m]$, the sum $\sum_{i \in S} M^i$ is denoted by M^S. We denote the $n \times n$ identity matrix by I_n. A real symmetric $n \times n$ matrix M is called *positive semi-definite* (PSD), denoted by $M \succeq 0$, if the scalar $z^T M z$ is non-negative for every $z \in \mathbb{R}^n$. The following conditions are well-known to be equivalent.

- **(i)** $M \succeq 0$
- **(ii)** Every eigenvalue of M is non-negative.
- **(iii)** $M = W^T W$ for some $n \times n$ matrix W. In other words, there exist n vectors $w_1, \ldots, w_n \in \mathbb{R}^n$ such that $M_{ij} = w_i^T w_j$.

Given two vectors $x, y \in \mathbb{R}^n$, their dot product $\langle x, y \rangle$ is defined to be $x^T y$. Given $M \succeq 0$, the inner product of x, y w.r.t. M, denoted by $\langle x, y \rangle_M$, is defined to be $x^T M y$. The usual dot product corresponds to the case $M = I$, the identity matrix. Every $n \times n$ symmetric matrix M has a spectral decomposition $M = U \Sigma U^T$, where the rows of U form an eigenbasis. If M has rank k, we can truncate the zero eigenvalues in Σ to obtain a truncated spectral decomposition. Now, Σ is a $k \times k$ diagonal matrix with the eigenvalues $\lambda_1, \ldots, \lambda_k \in \mathbb{R}$ on the diagonal. The matrix U is a $n \times k$ matrix with the corresponding eigenvectors $v_1, \ldots, v_k$ as the columns $U_1, \ldots, U_k$. We will always work with truncated spectral decompositions henceforth.

Frobenius Norm. The trace of a matrix M, denoted by $\mathrm{Tr}(M)$, is defined to be $\sum_{i \in [n]} M_{ii}$. The *trace inner product* of two matrices A and B, denoted by $\mathrm{Tr}(A, B)$, is the scalar $\mathrm{Tr}(A^T B)$. The *Frobenius norm* $\|M\|_F$ of a matrix M is defined to be $\sum_{i,j \in [n]} {M_{ij}}^2$. It is easy to check that $\|M\|_F^2 = \mathrm{Tr}(M, M)$. Given two n-vertex graphs G and H and a permutation $\pi \in S_n$, a *π-mismatch* between G and H is a pair $\{i, j\}$ such that $\{i, j\} \in E(G)$ and $\{i^\pi, j^\pi\} \notin E(H)$ (or vice-versa). In other words, $\pi : V(G) \to V(H)$ does not preserve adjacency for the pair $\{i, j\}$. The following claim will be useful as a combinatorial interpretation of the Frobenius norm. Let Δ denote the number of π-mismatches between G and H.

▶ **Claim 2.** $\|A_G^\pi - A_H\|_F^2 = 2\Delta$.

Proof. The only non-zero terms in the expansion of summation $\|A_G^\pi - A_H\|_F^2$ correspond to π-mismatches. Since every mismatch $\{i, j\}$ contributes 1 and is counted twice in the summation, the claim follows. ◀

Clustering Number. Spectral Graph Drawing is a well-established technique for visualizing graphs via their spectral properties (see e.g. [17]). We introduce the details necessary for our results. Let A be a $n \times n$ matrix of rank k. Let G be the corresponding adjacency graph, with the vertex set $[n]$. Given a spectral decomposition $A = U \Lambda U^T$, Σ is a $k \times k$ matrix and U is a $n \times k$ matrix. Since the spectral decomposition of a matrix is not unique, the following claim will be useful.

▶ **Claim 3.** *Given two spectral decompositions $A = U \Lambda U^T$ and $A = U' \Lambda {U'}^T$, the number of distinct elements in the multi-set* $\mathrm{rows}(U)$ *is equal to the number of distinct elements in the multi-set* $\mathrm{rows}(U')$.

Therefore, the number of *distinct* elements in the multi-set $\mathrm{rows}(U)$ is invariant of our choice of spectral decomposition $A = U \Lambda U^T$. This allows us to define the *clustering number* of a graph G, denoted by $\mathrm{cn}(G)$, as the number of *distinct* elements in the multi-set $\mathrm{rows}(U)$, for some spectral decomposition $A = U\Lambda U^T$. The clustering number of a matrix A, denoted by $\mathrm{cn}(A)$, is defined to be the clustering number of the corresponding adjacency graph.

Let $A = U\Lambda U^T$ be a PSD matrix. The following theorem relates the clustering number $\mathrm{cn}(A)$ to the number of twin-equivalence partitions $\tau(A)$.

▶ **Theorem 4.** *Let A be a PSD matrix. The number of twin-equivalence classes $\tau(A)$ is equal to p if and only if A has p distinct elements in the set* $\mathrm{rows}(U)$ *for a spectral decomposition $A = U\Lambda U^T$.*

Hyperplanes and Convex Functions. A *hyperplane* H in the Euclidean space $\mathbb{R}^k$ is a $(k-1)$-dimensional affine subspace. The usual representation of a hyperplane is a linear equation $\langle c, x\rangle = \alpha$ for some $c \in \mathbb{R}^k, \alpha \in \mathbb{R}$. The convex sets $\{x \mid \langle c, x\rangle > \alpha\}$ and $\{x \mid \langle c, x\rangle < \alpha\}$ are called the open *half-spaces* corresponding to H, denoted by H^+, H^- respectively.

Two sets (S, T) are *weakly linearly separated* if there exists a hyperplane H such that $S \subseteq H^+ \cup H$ and $T \subseteq H^- \cup H$. In this case, we call them weakly linearly separated along H. A family of sets $S_1, \ldots, S_p$ is *weakly linearly separated* if for every $l, m \in [p]$, the sets S_l, S_m are weakly linearly separated. Let Π be a partition of a set S into p sets $S_1, \ldots, S_p$. The partition Π is said to be *mutually linearly separated* if the family of sets $S_1, \ldots, S_p$ is weakly linearly separated.

Recall that a subset $S \subseteq \mathbb{R}^k$ is called *convex* if for every $x, y \in S$, $\alpha x + (1-\alpha)y \in S$, $\alpha \in [0, 1]$. A function $f : \mathbb{R}^k \to \mathbb{R}$ is called *convex* on a convex set S if for every $x, y \in S$, $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$. The following theorem about linearization of convex differentiable functions is well-known and is stated without proof. The gradient of a function $f : \mathbb{R}^k \to \mathbb{R}$, denoted by ∇f, is the vector-valued function $[\frac{\partial f}{\partial x_1} \ldots \frac{\partial f}{\partial x_k}]$. Given $X^* \in \mathbb{R}^k$, let μ^* denote the vector $\nabla f(X^*)$.

▶ **Theorem 5** (Convex function linearization). *Let $f : \mathbb{R}^k \to \mathbb{R}$ be a convex function. For all $X \in \mathbb{R}^k$, $f(X) - f(X^*) \geq \langle \mu^*, X - X^*\rangle$.*

Finally, we state an important fact about the convexity of quadratic functions. Given a PSD matrix $M \in \mathbb{R}^{k\times k}$, the quadratic function $Q_M : \mathbb{R}^k \to \mathbb{R}$ is defined as $Q_M(x) = \langle x, x\rangle_M$.

▶ **Lemma 6** (Convexity of PSD). *Q_M is convex on $\mathbb{R}^k$.*

3 Hardness Results

In this section, we show several new hardness results for problems GSim, WSim and MSim. As we will observe, these problems turn out to be algorithmically intractable, even for severely restricted cases. We begin by recalling the following observation.

▶ **Theorem 7** (Folklore). *GSim is NP-hard for the class of simple undirected graphs.*

In fact, the problem turns out to be NP-hard even for very restricted graph classes. The following theorem is the main hardness result of this section.

▶ **Theorem 8.** *GSim is NP-hard for the class of trees.*

Proof. The proof is by a reduction from the following NP-hard variant of the Three-Partition problem [12], which is defined as follows. The input consists of integers A and $a_1, \ldots, a_{3m}$ in unary representation, with $\sum_{i=1}^{3m} a_i = mA$ and with $A/4 < a_i < A/2$ for $1 \leq i \leq 3m$. The question is to decide whether $a_1, \ldots, a_{3m}$ can be partitioned into m triples so that the elements in each triple sum up to precisely A.

We first show that the restriction of GSim to forests is NP-hard. Given an instance of Three-Partition, we compute an instance of GSim on the following two forests F_1 and F_2. Forest F_1 is the disjoint union of $3m$ paths with $a_1, \ldots, a_{3m}$ vertices, respectively. Forest F_2 is the disjoint union of m paths that each consists of A vertices. We claim that the Three-Partition instance has answer YES, if and only if there exists a permutation π such that there are at most $2m$ mismatches. If the desired partition exists, then for each triple we we can pack the three corresponding paths in F_1 into one of the paths in F_2 with two mismatches per triple. Conversely, if there exists a permutation π with at most $2m$ mismatches, then these $2m$ mismatches cut the paths in F_2 into $3m$ subpaths (we consider isolated vertices as paths of length 0). As each of these $3m$ subpaths must be matched with a path in F_1, we easily deduce from this a solution for the Three-Partition instance.

To show that GSim is NP-hard for the class of trees, we modify the above forests F_1 and F_2 into trees T_1 and T_2. Formally, we add a new vertex v_1 to $V(F_1)$ and then connect one end-point of every path in F_1 to v_1 by an edge; note that the degree of vertex v_1 in the resulting tree is $3m$. Analogously, we add a new vertex v_2 to $V(F_2)$, connect it to all paths, and thus produce a tree in which vertex v_2 has degree m. For technical reasons, we furthermore attach $8m$ newly created leaves to every single vertex in $V(F_1)$ and $V(F_2)$. k The resulting trees are denoted T_1 and T_2, respectively.

We claim that the considered Three-Partition instance has answer YES, if and only if there exists $\pi : V(T_1) \to V(T_2)$ with at most $4m$ mismatches. If the desired partition exists, the natural bijection maps every original forest edge in T_1 to an original forest edge in T_2, except for some $2m$ out of the $3m$ edges that are incident to v_1 in T_1; this yields a total number of $2m + 2m = 4m$ mismatches. Conversely, suppose that there exists a permutation π with at most $4m$ mismatches. Then π must map v_1 in T_1 to v_2 in T_2, since otherwise we pay a penalty of more than $4m$ mismatches alone for the edges incident to the vertex mapped into v_2. As the number of mismatches for edges incident to v_1 and v_2 amounts to $2m$, there remain at most $2m$ further mismatches for the remaining edges. Similarly as in our above argument for the forests, these at most $2m$ mismatches yield a solution for the Three-Partition instance. ◀

On the other hand, if we restrict one of the input instances to be a path, the problem can be solved in polynomial time. The following theorem provides a positive example of tractability of GSim.

▶ **Theorem 9.** *An input instance (G, H) of* GSim*, where G is a path and H is a tree, can be solved in polynomial time.*

The above results exhibit the hardness of GSim, and consequently, the hardness of the more general problems WSim and MSim. Since the graphs (for instance cycles and paths) involved in the hardness reductions have adjacency matrices of high rank, it is natural to ask whether MSim would become tractable for matrices of low rank. Our following theorem shows that MSim is NP-hard even for matrices of rank at most 2. The underlying reason for hardness is the well-known problem QAP, which shares the optimization domain S_n.

▶ **Theorem 10.** MSim *is NP-hard for symmetric matrices of rank at most* 2*.*

The key to the above reduction is the fact that one of the matrices has non-negative eigenvalues while the other matrix has non-positive eigenvalues. We show that the MSim is NP-hard even for positive semi-definite matrices. The main idea is to reformulate the hardness reduction in Theorem 7 in terms of Laplacian matrices.

▶ **Theorem 11.** MSim *is NP-hard for positive semi-definite matrices.*

In fact, we show that the problem remains NP-hard, even if one of the matrices is of rank 1. The proof follows by modifying the matrices in the proof of Theorem 10 so that they are positive semi-definite.

▶ **Theorem 12.** MSim *is NP-hard for positive semi-definite matrices, even if one of the matrices has rank* 1*.*

Therefore, the realm of tractability for MSim is restricted to positive definite matrices of bounded rank.

4 The QVP Problem

We proceed towards the proof of Theorem 1, our main algorithmic result about MSim. In order to prove this theorem, we need to define an intermediate problem, called the Quadratic-Vector-Partition (QVP). In this section, we study several aspects of this problem. First, we state this problem, and show an efficient reduction from MSim to QVP (Sections 4.1 and 4.2). The definition of the problem QVP is slightly technical; the ensuing reduction, from MSim to QVP, will justify the introduction of this intermediate problem. In Sections 4.3 and 4.4, we will establish strong conditions on the optimal solutions for a QVP instance. Later on, in Section 5, these conditions will allow us to design efficient algorithms for QVP, which will finish the proof of Theorem 1.

4.1 QVP, definition

Let p and k be fixed positive integers. The *input instance* to QVP is a tuple (W, K, Λ, Δ), where

- W is a set of n vectors $\{w_1, \ldots, w_n\} \subseteq \mathbb{R}^k$,
- K is a $p \times p$ PSD matrix,
- Λ is a $k \times k$ *diagonal* matrix with non-negative entries, and,
- Δ is (the unary encoding of) a p-tuple $(n_1, \ldots, n_p)$ such that $n_1 + \cdots + n_p = n$.

Some additional notation is required, before we proceed further. An ordered partition $T_1 \dot\cup \cdots \dot\cup T_p$ of $[n]$ is said to have *type* Δ if the cardinalities $|T_l| = n_l$, for all $l \in [p]$. Let $\mathcal{P}_\Delta$ denote the set of all (ordered) partitions of $[n]$ of type Δ. Let T be a subset of $[n]$. Denote the subset of W indexed by the set T as $W[T] = \{w_j \,|\, j \in T\}$. The centroid of the subset $W[T]$ is denoted by $\widehat{w_T}$. In other words, $\widehat{w_T} = \frac{1}{|T|} \sum_{i \in T} w_i$.

We continue with the definition of QVP. Given a partition $P = (T_1, \ldots, T_p) \in \mathcal{P}_\Delta$, the QVP objective function $F(P)$ is defined as

$$F(P) = \sum_{l,m \in [p]} K_{lm} \left\langle \widehat{w}_{T_l}, \widehat{w}_{T_m} \right\rangle_\Lambda .$$

The optimization problem QVP is to compute a partition $P^* \in \mathcal{P}_\Delta$ which is a maximizer of the objective function $F(P)$ over the domain $\mathcal{P}_\Delta$.

4.2 MSim reduces to QVP

Let k and p be fixed positive integers. Let (A, B) be an MSim instance, as defined in Theorem 1: the PSD matrices A and B are of rank at most k, and moreover, $\tau(B) = p$. The following lemma describes a reduction from MSim to QVP. Here, the $\widetilde{\mathcal{O}}$ notation hides factors polynomial in the size of the input representation.

▶ **Lemma 13.** *There exists an $\widetilde{\mathcal{O}}(n^3)$ running time algorithm which can transform the* MSim *instance (A, B) into a* QVP*-instance (W, K, Λ, Δ), with the following property. Given an optimal solution for this* QVP*-instance, we can compute an optimal solution for the* MSim *instance, in $\mathcal{O}(n)$ running time.*

Proof. Fix two spectral decompositions $A = U\Lambda U^T$ and $B = V\Gamma V^T$ of A and B respectively. Since $\tau(B) = p$, the multiset rows(V) has exactly p distinct vectors (by Theorem 4). Let these p distinct vectors be denoted by $\{\tilde{V}^1, \ldots, \tilde{V}^p\}$. Let $n_1, \ldots, n_p$ be the multiplicity of the elements $\tilde{V}^1, \ldots, \tilde{V}^p$ in the multiset rows(V). Clearly, $n_1 + \cdots + n_p = n$. Let $\tilde{P} = S_1 \cup \cdots \cup S_p$ be the partition of the set $[n]$ such that $S_l = \{i \mid V^i = \tilde{V}^l\}$, for $l \in [p]$. In other words, the partition $\tilde{P}$ encodes the equivalence relation $V^i = V^j$, where $i, j \in [n]$.

Let us describe the polynomial time transformation of the MSim instance (A, B) into the QVP instance (W, K, Λ, Δ). Define W as the multiset rows(U). In other words, we can denote $W = \{w_1, \ldots, w_n\}$ where $w_i = U^i$. Define K to be the $p \times p$ matrix defined as $K_{lm} = |S_l| \cdot |S_m| \cdot \langle \tilde{V}^l, \tilde{V}^m \rangle_\Gamma$, for $l, m \in [p]$. Since we can write $K_{lm} = \langle |S_l| \cdot \tilde{V}^l, |S_m| \cdot \tilde{V}^m \rangle_\Gamma$, we can show that K is positive semi-definite. We set Λ to be the $k \times k$ diagonal matrix in the spectral decomposition $A = U\Lambda U^T$. Finally, we set Δ to be $(n_1, \ldots, n_p)$: these numbers were defined in the previous paragraph. The computation of this QVP instance can be performed in $\widetilde{\mathcal{O}}(n^3)$ time, which is the time taken to compute the spectral decompositions for A and B.

It remains to show that an optimal solution for this QVP instance yields an optimal solution for the MSim instance in $\mathcal{O}(n)$ time. Observe that $\|A^\pi - B\|_F^2 = \mathrm{Tr}(A^\pi - B, A^\pi - B) = \mathrm{Tr}(A^\pi, A^\pi) + \mathrm{Tr}(B, B) - 2\,\mathrm{Tr}(A^\pi, B)$. Since $\mathrm{Tr}(A^\pi, A^\pi) = \|A^\pi\|_F^2 = \|A\|_F^2 = \mathrm{Tr}(A, A)$, we have $\|A^\pi - B\|_F^2 = \mathrm{Tr}(A, A) + \mathrm{Tr}(B, B) - 2\,\mathrm{Tr}(A^\pi, B)$. This derivation implies that we can equivalently maximize $\mathrm{Tr}(A^\pi, B)$ over $\pi \in S_n$. Observe that $\mathrm{Tr}(A^\pi, B)$ can be rewritten as

$$\begin{aligned}
\mathrm{Tr}(A^\pi, B) &= \sum_{i,j \in [n]} a_{i^\pi j^\pi}\, b_{ij} \\
&= \sum_{i,j \in [n]} \langle U^{i^\pi}, U^{j^\pi} \rangle_\Lambda \cdot \langle V^i, V^j \rangle_\Gamma \\
&= \sum_{l,m \in [p]} \left(\sum_{i \in S_l,\, j \in S_m} \langle U^{i^\pi}, U^{j^\pi} \rangle_\Lambda \cdot \langle V^i, V^j \rangle_\Gamma \right)
\end{aligned}$$

which can be further re-written as

$$\begin{aligned}
\mathrm{Tr}(A^\pi, B) &= \sum_{l,m \in [p]} \left(\sum_{i \in S_l,\, j \in S_m} \langle U^{i^\pi}, U^{j^\pi} \rangle_\Lambda \right) \cdot \langle \tilde{V}^l, \tilde{V}^m \rangle_\Gamma \\
&= \sum_{l,m \in [p]} \left\langle \sum_{i \in S_l} U^{i^\pi}, \sum_{j \in S_m} U^{j^\pi} \right\rangle_\Lambda \cdot \langle \tilde{V}^l, \tilde{V}^m \rangle_\Gamma \\
&= \sum_{l,m \in [p]} |S_l| \cdot |S_m| \cdot \left\langle \widehat{w}_{S_l^\pi}, \widehat{w}_{S_m^\pi} \right\rangle_\Lambda \cdot \langle \tilde{V}^l, \tilde{V}^m \rangle_\Gamma
\end{aligned}$$

Define the partition P_π of $[n]$ to be $P_\pi = (S_1^\pi, \ldots, S_p^\pi)$. Observe that P_π is of type Δ, and therefore, $P_\pi \in \mathcal{P}_\Delta$. Using the definition of the matrix K, we can thus rewrite

$$\begin{aligned}
\operatorname{Tr}(A^\pi, B) &= \sum_{l,m \in [p]} |S_l| \cdot |S_m| \cdot \left\langle \widehat{w}_{S_l^\pi}, \widehat{w}_{S_m^\pi} \right\rangle_\Lambda \cdot \langle \tilde{V}^l, \tilde{V}^m \rangle_\Gamma \\
&= \sum_{l,m \in [p]} \left\langle \widehat{w}_{S_l^\pi}, \widehat{w}_{S_m^\pi} \right\rangle_\Lambda \cdot K_{lm} \\
&= F(P_\pi),
\end{aligned}$$

which allows us to state the following equality.

$$\|A^\pi - B\|_F^2 = \operatorname{Tr}(A, A) + \operatorname{Tr}(B, B) - 2F(P_\pi). \tag{2}$$

We continue with the proof of the lemma. Let P^* be an optimal solution for our QVP instance. In other words, the partition $P^* = (T_1^*, \ldots, T_p^*)$ is a maximizer of $F(P)$ over the set $\mathcal{P}_\Delta$. Let π^* be a permutation which maps the sets S_l to T_l^*, for all $l \in [p]$. We claim that π^* is an optimal solution for the MSim instance. To see this, suppose π^* is not optimal. Instead, let π' be an optimal solution for the MSim instance, and hence, $\|A^{\pi^*} - B\|_F^2 > \|A^{\pi'} - B\|_F^2$. Define a related partition $P_{\pi'} = (S_1^{\pi'}, \ldots, S_p^{\pi'})$: clearly, $\pi' \in \mathcal{P}_\Delta$. Since Equation 2 implies that

$$\begin{aligned}
\|A^{\pi^*} - B\|_F^2 &= \operatorname{Tr}(A, A) + \operatorname{Tr}(B, B) - 2F(P^*), \\
\|A^{\pi'} - B\|_F^2 &= \operatorname{Tr}(A, A) + \operatorname{Tr}(B, B) - 2F(P_{\pi'}),
\end{aligned}$$

we use $\|A^{\pi^*} - B\|_F^2 > \|A^{\pi'} - B\|_F^2$ to obtain that $F(P^*) < F(P_{\pi'})$. This contradicts the maximality of P^*. Hence, π^* must be an optimal solution for the QVP instance.

Given such an optimal solution P^* for the QVP instance, the computation of the optimal solution π^* for the MSim instance is a straightforward $\widetilde{\mathcal{O}}(n)$ procedure: we define π^* by choosing arbitrary bijections between the sets S_l and T_l^*, for all $l \in [p]$. This finishes the proof of our lemma. ◀

4.3 Linearization of Convex Functions

We take a small detour towards the properties of convex functions. These properties will be useful for studying the optimal solutions to the QVP problem. In general, we show that the linearization of a convex function can be useful in understanding its optima over a finite domain. In this context, we prove the following lemma about convex functions, which is interesting in its own right.

► **Lemma 14.** *Let Ω be a finite subset of $\mathbb{R}^k \times \mathbb{R}^\ell$. Let $G : \mathbb{R}^k \to \mathbb{R}$, $H : \mathbb{R}^\ell \to \mathbb{R}$ such that H is convex, and let $F : \mathbb{R}^k \times \mathbb{R}^\ell \to \mathbb{R}$ be defined as $F(X, Y) = G(X) + H(Y)$. Let $(X^*, Y^*) \in \arg\max_{(X,Y) \in \Omega} F(X, Y)$.*

Then there exist a $\mu^* \in \mathbb{R}^\ell$ such that:

(i) *$(X^*, Y^*) \in \arg\max_{(X,Y) \in \Omega} L(X, Y)$ where $L(X, Y) = G(X) + \langle \mu^*, Y \rangle$;*

(ii) *$\arg\max_{(X,Y) \in \Omega} L(X, Y) \subseteq \arg\max_{(X,Y) \in \Omega} F(X, Y)$.*

Proof. Let $(X^*, Y^*) \in \arg\max_{S \in \Omega} F(S)$. Since H is convex, we can use Theorem 5 to linearize H around $Y^* \in \mathbb{R}^\ell$. Hence, there exists a $\mu^* \in \mathbb{R}^\ell$ such that $H(Y) - H(Y^*) \geq \langle \mu^*, Y - Y^* \rangle$, or equivalently,

$$H(Y) - \langle \mu^*, Y \rangle \geq H(Y^*) - \langle \mu^*, Y^* \rangle, \tag{3}$$

for all $Y \in \mathbb{R}^\ell$. Hence with $L(X,Y) = G(X) + \langle \mu^*, Y\rangle$, for all $(X,Y) \in \Omega$ we have

$$L(X^*,Y^*) = F(X^*,Y^*) - H(Y^*) + \langle \mu^*, Y^*\rangle \geq F(X,Y) - H(Y) + \langle \mu^*, Y\rangle = L(X,Y),$$

where the inequality holds by (3) and because (X^*,Y^*) maximizes F. Hence (X^*,Y^*) maximizes L as well, which proves (i).

For (ii), consider $(X^{**},Y^{**}) \in \arg\max_{(X,Y)\in\Omega} L(X,Y)$. To prove that $(X^{**},Y^{**}) \in \arg\max_{(X,Y)\in\Omega} F(X,Y)$, it suffices to prove that $F(X^{**},Y^{**}) \geq F(X^*,Y^*)$. By (i), we have $L(X^*,Y^*) = L(X^{**},Y^{**})$. Thus

$$\begin{aligned} F(X^{**},Y^{**}) = L(X^{**},Y^{**}) + H(Y^{**}) - \langle \mu^*, Y^{**}\rangle &\geq L(X^*,Y^*) + H(Y^*) - \langle \mu^*, Y^*\rangle \\ &= F(X^*,Y^*), \end{aligned}$$

where the inequality holds by (3) with $(X,Y) := (X^{**},Y^{**})$ and as (X^{**},Y^{**}) maximizes L. ◀

In other words, for every (X^*,Y^*) which maximizes F over Ω, there exists a partially "linearized" function L such that (X^*,Y^*) maximizes L over Ω. Moreover, every maximizer of L over Ω is a maximizer of F over Ω. This additional condition is necessary so that this "linearization" does not create spurious optimal solutions.

▶ **Corollary 15.** *Let Ω be a finite subset of $\mathbb{R}^{kp}$. For all $i \in [k]$, let $G_i : \mathbb{R}^k \to \mathbb{R}$ be a convex function. Let $F : \mathbb{R}^{kp} \to \mathbb{R}$ be defined as $F(X_1,\dots,X_k) := G_1(X_1) + \dots + G_k(X_k)$. Let $X^* = (X_1^*,\dots,X_k^*) \in \arg\max_{X\in\Omega} F(X)$.*

Then there are $\mu_1^*,\dots,\mu_k^* \in \mathbb{R}^p$ such that:

(i) *$X^* \in \arg\max_{X\in\Omega} L(X)$ where $L(X_1,\dots,X_k) = \sum_{i=1}^k \langle \mu_i^*, X_i\rangle$;*

(ii) *$\arg\max_{X\in\Omega} L(X) \subseteq \arg\max_{\in\Omega} F(X)$.*

Proof. Inductively apply the lemma to the functions

$$F^i((X_1,\dots,X_{i-1},X_{i+1},\dots,X_k),X_i) = \underbrace{\left(\sum_{j=1}^{i-1}\langle \mu_j^*, X_j\rangle + \sum_{j=i+1}^{k} G_j(X_j)\right)}_{=:G^i(X_1,\dots,X_{i-1},X_{i+1},\dots,X_k)} + \underbrace{G_i(X_i)}_{=:H^i(X_i)}. \quad ◀$$

4.4 Optimal Solution Structure for QVP

Let us express the QVP objective function

$$F(P) = \sum_{l,m\in[p]} K_{lm}\,\langle \widehat{w}_{T_l}, \widehat{w}_{T_m}\rangle_\Lambda$$

as a restriction of a convex function to a finite domain. Using the results above for linearization of convex functions, we show that the optimal solutions for a QVP instance must satisfy certain structural constraints, specified by Lemma 19.

Formally, given a QVP instance (W,K,Λ,Δ), and a partition $P = (T_1,\dots,T_p) \in \mathcal{P}_\Delta$, we define k vectors $X_1,\dots,X_k$ as follows. For $q \in [k]$, let X_q be the vector of length p corresponding to the q^{th} coordinates of vectors $\widehat{w}_{T_1},\dots,\widehat{w}_{T_p}$. Clearly, the vectors $X_1,\dots,X_k$ are a function of the partition P. Recall that Λ is a diagonal matrix with k non-negative entries, say $\lambda_1,\dots,\lambda_k$.

▶ **Claim 16.** $F(P) = \displaystyle\sum_{q=1}^{k} \lambda_q \langle X_q, X_q\rangle_K$.

Observe that the function $G : \mathbb{R}^p \mapsto \mathbb{R}$ defined by $G(Y) = \langle Y, Y \rangle_K$ is a convex function (by Lemma 6). Define a function $\tilde{F}(Y_1, \ldots, Y_p) = \lambda_1 G(Y_1) + \cdots + \lambda_k G(Y_k)$, where the vectors $Y_1, \ldots, Y_k \in \mathbb{R}^p$. This function is convex as well: it is a linear combination of convex functions, with positive co-efficients. Observe that $F(P) = \tilde{F}(X_1, \ldots, X_p)$. Therefore, the problem of maximizing F over $\mathcal{P}_\Delta$ is, essentially, a problem of maximizing a convex function $\tilde{F}(Y_1, \ldots, Y_k)$ over a finite discrete domain.

Using Corollary 15, we claim that a maximizer $P^* = (T_1^*, \ldots, T_p^*)$ of the objective function $F(P)$ over the domain $\mathcal{P}_\Delta$ must be a maximizer for some *linear* objective function $L_1(P)$ over the domain $\mathcal{P}_\Delta$.

▶ **Claim 17.** *There exist vectors $\mu_1^*, \ldots, \mu_k^* \in \mathbb{R}^p$ such that P^* is a maximizer of the function $L_1(P) = \sum_{q=1}^{k} \lambda_q \langle \mu_q^*, X_q \rangle_K$ over $\mathcal{P}_\Delta$. Moreover, every maximizer of $L_1(P)$ is a maximizer of $F(P)$.*

We can further reformulate the optimality conditions of Claim 17 as follows.

▶ **Claim 18.** *There exist vectors $\mu_1, \ldots, \mu_p \in \mathbb{R}^k$ such that P^* is a maximizer of the function $L_2(P) = \sum_{m=1}^{p} \langle \mu_m, \widehat{w}_{T_m} \rangle$ over $\mathcal{P}_\Delta$. Moreover, every maximizer of $L_2(P)$ is a maximizer of $L_1(P)$, and consequently, a maximizer of $F(P)$.*

The above claim leads to a strong geometrical restriction on the partition $W[T_1^*], \ldots, W[T_p^*]$ of W, induced by the optimal partition $P^* \in \mathcal{P}_\Delta$.

▶ **Lemma 19.** *Let $P^* = (T_1^*, \ldots, T_p^*)$ be an optimal solution for a* QVP *instance (W, K, Λ, Δ). The partition $(W[T_1^*], \ldots, W[T_p^*])$ of the set W is (weakly) mutually linearly separated.*

Proof. By Claim 18, there exist vectors $\mu_1, \ldots, \mu_p \in \mathbb{R}^k$ such that P^* is a maximizer of $L_2(P)$. Suppose there exist q, r such that $W[T_q^*]$ and $W[T_r^*]$ are not (weakly) linearly separated. We claim that this leads to a contradiction.

Let n_q and n_r be the cardinalities of the sets T_q^* and T_r^*. We use the notation W^T to denote the sum of the vectors in the set $W[T]$, for a subset $T \subseteq [n]$. Let us isolate the two terms $\langle \mu_q, \widehat{w}_{T_q^*} \rangle + \langle \mu_r, \widehat{w}_{T_r^*} \rangle$ and rewrite them as $\langle \frac{\mu_q}{n_q}, W^{T_q^*} \rangle + \langle \frac{\mu_r}{n_r}, W^{T_r^*} \rangle$. Let us denote $\frac{1}{n_q}\mu_q$ by μ_q', and $\frac{1}{n_r}\mu_r$ by μ_r'. Therefore, the terms can be re-written as $\langle \mu_q', W^{T_q^*} \rangle + \langle \mu_r', W^{T_r^*} \rangle$.

Rewriting further, we can express the above terms as $\langle (\mu_q' - \mu_r'), W^{T_q^*} \rangle + \langle \mu_r', (W^{T_q^*} + W^{T_r^*}) \rangle$. Recall that the sets $W[T_q^*]$ and $W[T_r^*]$ are not weakly linearly separated. Let us partition the set $W[T_q^*] \cup W[T_r^*]$ into two sets $W[T_q']$ and $W[T_r']$ such that (a) $|T_q'| = n_q$, $|T_r'| = n_r$ and (b) the sets $W[T_q']$ and $W[T_r']$ are weakly linearly separated along the direction $(\mu_q' - \mu_r')$. Indeed, we can sort all elements w in $W[T_q^*] \cup W[T_r^*]$ in a descending order, according to their (signed) projection $\langle (\mu_q' - \mu_r'), w \rangle$ along $(\mu_q' - \mu_r')$. Pick the top n_q elements in this ordering to obtain the set T_q' and collect the remaining n_r elements to form the set T_r'. Note that the sets T_q' and T_r' are weakly linearly separated along the direction $(\mu_q' - \mu_r')$, and hence, the pair $(T_q^*, T_r^*) \neq (T_q', T_r')$.

Clearly, $\langle (\mu_q' - \mu_r'), W^{T_q'} \rangle > \langle (\mu_q' - \mu_r'), W^{T_q^*} \rangle$ by our construction. Moreover, $\langle \mu_r', (W^{T_q'} + W^{T_r'}) \rangle = \langle \mu_r', (W^{T_q^*} + W^{T_r^*}) \rangle$, because $T_q' \cup T_r' = T_q^* \cup T_r^*$. Hence, $\langle \mu_q', W^{T_q'} \rangle + \langle \mu_r', W^{T_r'} \rangle > \langle \mu_q', W^{T_q^*} \rangle + \langle \mu_r', W^{T_r^*} \rangle$, which implies that $\langle \mu_q, \widehat{w}_{T_q'} \rangle + \langle \mu_r, \widehat{w}_{T_r'} \rangle > \langle \mu_q, \widehat{w}_{T_q^*} \rangle + \langle \mu_r, \widehat{w}_{T_r^*} \rangle$. This contradicts the maximality of P^* for the function $L_2(P)$ over the domain $\mathcal{P}_\Delta$. ◀

5 Proof of Theorem 1

In this section, we prove the following algorithmic result about QVP.

▶ **Theorem 20.** *Given a* QVP *instance* (W, K, Λ, Δ)*, we can compute an optimal solution for this instance in* $\widetilde{\mathcal{O}}(n^{kp^2})$ *time.*

In this section, we will prove Theorem 20 in a restricted setting: we assume that the set W is in General Position (G.P.). The proof for the general case is not very different: using a technical tool to handle degeneracies in W, we can reduce the general case to this restricted case. We defer the proof of Theorem 20 (i.e., the general case) to the full version of the paper, and continue with the proof for this restricted setting.

Observe that the proof of Theorem 1 follows immediately from the above theorem.

Proof of Theorem 1. Let (A, B) be an MSIM instance, as defined in the statement of Theorem 1. Using the reduction in Lemma 13, we can compute a QVP instance (W, K, Λ, Δ) in $\widetilde{\mathcal{O}}(n^3)$ with the following property: an optimal solution to this QVP instance can be used to compute an optimal solution for the MSIM instance (A, B), in $\mathcal{O}(n)$ time. Using Theorem 20, we can compute an optimal solution for the QVP instance (W, K, Λ, Δ) in $\widetilde{\mathcal{O}}(n^{kp^2})$ time. Therefore, we can compute an optimal solution for the MSIM instance in overall $\widetilde{\mathcal{O}}(n^{kp^2})$ time. ◀

5.1 Algorithm for QVP, restricted version

We proceed with the proof of Theorem 20, under the G.P. assumption. In other words, given a QVP instance (W, K, Λ, Δ), the input set W is in General Position. Recall that a set S of n points $w_1, \dots, w_n \in \mathbb{R}^k$ is said to be in *general position* (G.P.), if there is no subset $S' \subseteq S$ with $|S'| > k$ that lies on a common hyperplane. Moreover, we can associate a unique hyperplane H_S with every k-element subset S of W. Let $\mathcal{H}$ be the set of $\binom{n}{k}$ hyperplanes, defined by each k-element subset of W. Under the G.P. assumption, we can further strengthen Lemma 19, as follows.

▶ **Lemma 21.** *Let* $P^* = (T_1, \dots, T_p)$ *be an optimal solution for a* QVP *instance* (W, K, Λ, Δ)*. For every pair of sets* $W[T_i]$ *and* $W[T_j]$*, where* $i < j$*, there exists a hyperplane* H_{ij} *in the set* $\mathcal{H}$ *such that* $W[T_i]$ *and* $W[T_j]$ *are weakly linearly separated along* H_{ij}*.*

The proof of this lemma follows immediately from the following claim.

▶ **Claim 22.** *Let* W *be a set of* n *points* $\{w_1, \dots, w_n\} \subset \mathbb{R}^k$ *in general position, where* $n > k$*. Suppose* W_1, W_2 *are two disjoint subsets of* W *which are weakly linearly separated by a hyperplane* H*. Then, there exists another hyperplane* $\tilde{H}$ *with the following properties: (a)* $\tilde{H}$ *passes through exactly* k *points of* W*, and (b)* $\tilde{H}$ *also weakly linearly separates* W_1, W_2*.*

Enumerative Algorithm. We proceed with an informal description of the algorithm. The overall strategy of our algorithm follows from Lemma 19 and Lemma 21. We will enumerate a particular subset $\mathcal{P}$ of $\mathcal{P}_\Delta$ defined as follows. The set $\mathcal{P}$ is the set of all weakly linearly separated partitions $P = (T_1, \dots, T_p)$ of W with the following property (stated in Lemma 21). For every pair of sets $W[T_i]$ and $W[T_j]$, where $i < j$, there exists a hyperplane H_{ij} in $\mathcal{H}$ such that $W[T_i]$ and $W[T_j]$ are weakly linearly separated along H_{ij}. Clearly, we can maximize the objective function $F(P)$ *over the set* $\mathcal{P}$, instead of the original domain $\mathcal{P}_\Delta$:

by Lemma 21, an optimal solution must lie in the set $\mathcal{P}$. Therefore, it suffices to prove the following lemma.

▶ **Lemma 23** (Enumeration, under G.P. assumption). *Given a* QVP *instance* (W, K, Λ, Δ), *assume that the set* W *is in General Position. Then, we can enumerate the set* $\mathcal{P}$ *in* $\widetilde{\mathcal{O}}(n^{kp^2})$ *time.*

Proof. From Lemma 21, we can deduce that a partition $P = (T_1, \ldots, T_p) \in \mathcal{P}$ can be associated with a sequence of $\binom{p}{2}$ separating hyperplanes $H_{ij} \in \mathcal{H}$, $i < j$, $i, j \in [p]$. In particular, the hyperplane H_{ij} weakly linearly separates $W[T_i]$ and $W[T_j]$.

Therefore, we enumerate the set $\mathcal{P}$ as follows. We branch over every choice of $|\mathcal{H}|^{\binom{p}{2}} \leq n^{\frac{kp^2}{2}}$ sequences of hyperplanes. We can define p convex regions $R_1, \ldots, R_p$ using these hyperplanes; the region R_i is supposed to contain the set $W[T_i]$, $i \in [p]$.

We assign the elements of W to these p disjoint convex regions $R_1, \ldots, R_p$. It is possible that an element w_j might lie on one or more of the hyperplanes H_{ij}. For such an 'ambigious' point, we brute-force try all possible p assignments of regions R_i. Since every hyperplane in $\mathcal{H}$ contains exactly k points of W, there can be at most $\binom{p}{2} \cdot k$ such ambigious points: this leads to an additional branching factor of at most $p^{\binom{p}{2} \cdot k}$. For each such branch, we obtain a partition $(W_1, \ldots, W_p)$ of W. If the type of this partition is not equal to Δ, we reject it; otherwise we add it to the list $\mathcal{P}$. The overall branching is bounded by $n^{\frac{kp^2}{2}} \cdot p^{\binom{p}{2} \cdot k}$ which is bounded by $n^{\frac{kp^2}{2}} \cdot n^{\frac{kp^2}{2}} \leq n^{kp^2}$. The overall running time is bounded by $\widetilde{\mathcal{O}}(n^{kp^2})$.

Clearly, every partition P in $\mathcal{P}$ can be discovered along some branch of our computation: we branch over all hyperplane sequences and further, over all assignments of 'ambigious' points. Moreover, every partition enumerated above belongs to $\mathcal{P}$, by our construction. Our overall branching factor of $\widetilde{\mathcal{O}}(n^{kp^2})$ is also an upper bound on the cardinality of $\mathcal{P}$. This finishes the proof of the lemma. ◀

Since we can enumerate the set $\mathcal{P}$ in $\widetilde{\mathcal{O}}(n^{kp^2})$ time, the optimal solution can be computed in a similar time as well. We summarize the above discussion as the following theorem.

▶ **Theorem 24** (QVP algorithm, G.P. assumption). QVP *can be solved in* $\widetilde{\mathcal{O}}(n^{kp^2})$ *running time.*

6 Conclusion

Through our results, we were able to gain insight into the tractibility of the problems GSim and MSim. However, there are a few open threads which remain elusive. The regime of bounded rank k and unbounded parameter $\tau(G)$ is still not fully understood for MSim, in the case of positive semi-definite matrices. It is not clear whether the problem is P-time or NP-hard in this case. Indeed, an $n^{O(k)}$ algorithm for MSim, in the case of positive semi-definite matrices, remains a possibility. From the perspective of parameterized complexity, we can ask if MSim is W[1]-hard, where the parameter of interest is the rank k. Finally, the approximability for the problems MSim deserves further examination, especially for the case of bounded rank.

References

1 H.A. Almohamad and S.O. Duffuaa. A linear programming approach for the weighted graph matching problem. *IEEE Transactions on pattern analysis and machine intelligence*, 15(5):522–525, 1993.

2 S. Arora, A. Frieze, and H. Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. *Mathematical programming*, 92(1):1–36, 2002.

3 V. Arvind, J. Köbler, S. Kuhnert, and Y. Vasudev. Approximate graph isomorphism. In B. Rovan, V. Sassone, and P. Widmayer, editors, *Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science*, volume 7464 of *Lecture Notes in Computer Science*, pages 100–111. Springer Verlag, 2012.

4 L. Babai. Graph isomorphism in quasipolynomial time. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC '16)*, pages 684–697, 2016.

5 R.E. Burkard, E. Cela, G. Rote, and G.J. Woeginger. The quadratic assignment problem with a monotone anti-monge and a symmetric toeplitz matrix: easy and hard cases. *Mathematical Programming*, 82:125–158, 1998.

6 E. Cela. *The Quadratic Assignment Problem: Theory and Algorithms*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998.

7 E. Cela, V.G. Deineko, and G.J. Woeginger. Well-solvable cases of the qap with block-structured matrices. *Discrete Applied Mathematics*, 186:56–65, 2015.

8 E. Cela, N. Schmuck, S. Wimer, and G.J. Woeginger. The wiener maximum quadratic assignment problem. *Discrete Optimization*, 8:411–416, 2011.

9 P. Codenotti, H. Katebi, K. A. Sakallah, and I. L. Markov. Conflict analysis and branching heuristics in the search for graph automorphisms. In *2013 IEEE 25th International Conference on Tools with Artificial Intelligence, Herndon, VA, USA, November 4-6, 2013*, pages 907–914, 2013.

10 D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 18(3):265–298, 2004.

11 A.N. Elshafei. Hospital layout as a quadratic assignment problem. *Operational Research Quarterly*, 28:167–179, 1977.

12 M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

13 A.M. Geoffrion and G.W. Graves. Scheduling parallel production lines with changeover costs: Practical application of a quadratic assignment/lp approach. *Operational Research*, 24:595–610, 1976.

14 S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on pattern analysis and machine intelligence*, 18(4):377–388, 1996.

15 T. Junttila and P. Kaski. Engineering an efficient canonical labeling tool for large and sparse graphs. In *Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments and the Fourth Workshop on Analytic Algorithms and Combinatorics*, pages 135–149. SIAM, 2007.

16 P. Keldenich. Random robust graph isomorphism. Master's thesis, Department of Compter Science, RWTH Aachen University, 2015.

17 Y. Koren. Drawing graphs by eigenvectors: theory and practice. *Computers and Mathematics with Applications*, 49(11):1867–1888, 2005.

18 J. Krarup and Pruzan P.M. Computer-aided layout design. *Mathematical Programming Study*, 9:75–94, 1978.

19 K. Makarychev, R. Manokaran, and M. Sviridenko. Maximum quadratic assignment problem: Reduction from maximum label cover and lp-based approximation algorithm. *ACM Transactions on Algorithms*, 10(4):18, 2014.

20 D.W. Matula. Subtree isomorphism in $o(n^{5/2})$. In P. H. B. Alspach and D. Miller, editors, *Algorithmic Aspects of Combinatorics*, volume 2 of *Annals of Discrete Mathematics*, pages 91–106. Elsevier, 1978.

21 B. McKay. Practical graph isomorphism. *Congressus Numerantium*, 30:45–87, 1981.

22 B. D. McKay and A. Piperno. Practical graph isomorphism, II. *J. Symb. Comput.*, 60:94–112, 2014.

23 S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings. 18th International Conference on Data Engineering*, pages 117–128, 2002.

24 V. Nagarajan and M. Sviridenko. On the maximum quadratic assignment problem. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 516–524, 2009.

25 D. Neuen and P. Schweitzer. Benchmark graphs for practical graph isomorphism. *ArXiv (CoRR)*, arXiv:1705.03686 [cs.DS], 2017.

26 R. O'Donnell, J. Wright, C. Wu, and Y. Zhou. Hardness of robust graph isomorphism, Lasserre gaps, and asymmetry of random graphs. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1659–1677, 2014.

27 M.A. Pollatschek, N. Gershoni, and Y.T. Radday. Optimization of the typewriter keyboard by simulation. *Angewandte Informatik*, 17:438–439, 1976.

28 F. Rendl and H. Wolkowicz. Applications of parametric programming and Eigenvalue maximization to the quadratic assignment problem. *Mathematical Programming*, 53:63–78, 1992.

29 S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE transactions on pattern analysis and machine intelligence*, 10(5):695–703, 1988.

30 M. Zaslavskiy, F. Bach, and J.-P. Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227–2242, 2009.