

MIT Open Access Articles

Spectral Alignment of Graphs

The MIT Faculty has made this article openly available. ***Please share*** how this access benefits you. Your story matters.

As Published: 10.1109/TNSE.2019.2913233

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: <https://hdl.handle.net/1721.1/133666>

Version: Original manuscript: author's manuscript prior to formal peer review

Terms of use: Creative Commons Attribution-Noncommercial-Share Alike



Spectral Alignment of Graphs

Soheil Feizi¹, Gerald Quon², Mariana Recamonde-Mendoza³, Muriel Médard⁴,
Manolis Kellis⁴ and Ali Jadbabaie⁴

Abstract

Graph alignment refers to the problem of finding a bijective mapping across vertices of two graphs such that, if two nodes are connected in the first graph, their images are connected in the second graph. This problem arises in many fields such as computational biology, social sciences, and computer vision and is often cast as a quadratic assignment problem (QAP). Most standard graph alignment methods consider an optimization that maximizes the number of matches between the two graphs, ignoring the effect of mismatches. We propose a generalized graph alignment formulation that considers both matches and mismatches in a standard QAP formulation. This modification can have a major impact in aligning graphs with different sizes and heterogenous edge densities. Moreover, we propose two methods for solving the generalized graph alignment problem based on spectral decomposition of matrices. We compare the performance of proposed methods with some existing graph alignment algorithms including Natalie2, GHOST, IsoRank, NetAlign, Klau’s approach as well as a semidefinite programming-based method over various synthetic and real graph models. Our proposed method based on simultaneous alignment of multiple eigenvectors leads to consistently good performance in different graph models. In particular, in the alignment of regular graph structures which is one of the most difficult graph alignment cases, our proposed method significantly outperforms other methods.

1 Introduction

The term graph alignment (or, network alignment) encompasses several distinct but related problem variants [1]. In general, graph alignment aims to find a bijective mapping across two (or more) graphs so that, if two nodes are connected in one graph, their images are also connected in the other graph(s). If such an exact alignment scheme exists, graph alignment can be simplified to the problem of graph isomorphism [2]. However, in general, an errorless alignment scheme may not be feasible. In such cases, graph alignment aims to find a mapping with the minimum error and/or the maximum overlap.

Graph alignment has a broad range of applications in systems biology, social sciences, computer vision, and linguistics. For instance, graph alignment has been used frequently as a comparative analysis tool in studying protein-protein interaction networks across different species [3–8]. In computer vision, graph alignment has been used for image recognition by matching similar images [9, 10]. It has also been applied in ontology alignment to find relationships among different representations of a database [11, 12], and in user de-anonymization to infer user/sample identifications using similarity between datasets [13].

¹ Stanford University.

² University of California, Davis.

³ Instituto de Informatica, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brazil.

⁴ Massachusetts Institute of Technology (MIT).

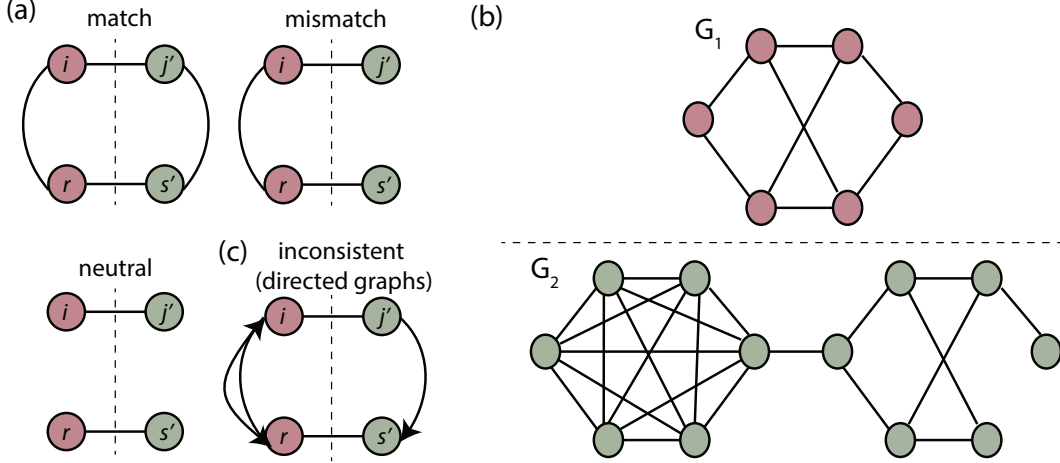


Figure 1: (a) An illustration of matched, mismatched, and neutral mappings for undirected graphs. (b) Example graphs to illustrate the effect of considering mismatches in the graph alignment formulation. (c) An illustration of inconsistent mappings for directed graphs where they are matches in one direction and mismatches in the other direction.

Here we study the graph alignment problem and make two main contributions. First, we propose a generalized formulation for the graph alignment optimization, and secondly we develop two graph alignment methods based on spectral decomposition of matrices. In the following we explain these contributions.

Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs where V_a and E_a represent set of nodes and edges of graph $a = 1, 2$, respectively. By a slight abuse of notation, let G_1 and G_2 be their matrix representations where for $a = 1, 2$, $G_a(i, j) = 1$ iff $(i, j) \in E_a$, and $G_a(i, j) = 0$ otherwise. Suppose graph a has n_a nodes, i.e., $|V_a| = n_a$. Without loss of generality, we assume $n_1 \leq n_2$. In the beginning, we assume graphs are undirected (i.e., matrices G_1 and G_2 are symmetric). We discuss the alignment of directed graphs, denoted by G_1^{dir} and G_2^{dir} , in Section 3.

Let X be an $n_1 \times n_2$ binary matrix where $X(i, j') = 1$ means that node i in graph 1 is mapped (aligned) to node j' in graph 2. The pair (i, j') is called a mapping edge across two graphs and is denoted by $i \leftrightarrow j'$. In the graph alignment setup, each node in one graph can be mapped to at most one node in the other graph, i.e., $\sum_i X(i, j') \leq 1$ for all j' , and similarly $\sum_{j'} X(i, j') \leq 1$ for all i . We also assume that there are no unaligned nodes in the graph with fewer nodes.

Matrix X can map an edge in graph G_1 to an edge in graph G_2 . These aligned edges are called *matches*. X can map an edge in G_1 to a non-existing edge in G_2 and vice versa. These mapping pairs are called *mismatches*. Finally, X can map a non-existing edge in G_1 to a non-existing edge in G_2 . These pairs are called *neutrals*. Figure 1-a illustrates examples of matches, mismatches, and neutrals for simple graphs with two nodes. We have

$$\begin{aligned}
 \# \text{ of matches} &= \text{Tr}(G_1 X G_2 X^T), \\
 \# \text{ of mismatches} &= \text{Tr}(G_1 X (\mathbb{1} - G_2) X^T + G_1 X (\mathbb{1} - G_2) X^T), \\
 \# \text{ of neutrals} &= \text{Tr}((\mathbb{1} - G_1) X (\mathbb{1} - G_2) X^T),
 \end{aligned} \tag{1.1}$$

where $\mathbb{1}$ represents a matrix of all ones and $\text{Tr}(\cdot)$ is the trace operator. Most existing scalable

graph alignment methods only consider maximizing the number of matches across two graphs while ignoring the number of resulting mismatches. This limitation can be critical particularly in cases where graphs have different sizes. We propose a generalized objective function for the graph alignment optimization as follows:

$$\begin{aligned} \max_X \quad & s_1(\# \text{ of matches}) + s_2(\# \text{ of neutrals}) \\ & + s_3(\# \text{ of mismatches}), \end{aligned} \tag{1.2}$$

where s_1 , s_2 , and s_3 are scores assigned to matches, neutrals, and mismatches, respectively. We assume $s_1 > s_2 > s_3$. Considering $s_2 = s_3 = 0$ results in ignoring effects of mismatches and neutrals. Substituting (1.1) in (1.2), we have the following equivalent optimization:

$$\max_X \quad \text{Tr}(G_1 X G_2 X^T) - \gamma (\text{Tr}(G_1 X \mathbb{1} X^T) + \text{Tr}(\mathbb{1} X G_2 X^T)), \tag{1.3}$$

where $\gamma = (s_2 - s_3)/(s_1 + s_2 - 2s_3)$ is the regularization parameter. Note $0 \leq \gamma < 1/2$. If $s_2 = s_3 = 0$, $\gamma = 0$, while if $s_2 \rightarrow s_1$ or $s_3 \rightarrow -\infty$, $\gamma \rightarrow 1/2$. If $n_1 = n_2$, $\text{Tr}(G_1 X \mathbb{1} X^T)$ and $\text{Tr}(\mathbb{1} X G_2 X^T)$ are equal to the number of edges in graphs G_1 and G_2 , respectively. Thus these terms do not depend on X . However, if the number of nodes in G_1 and G_2 are different (say $n_1 < n_2$), $\text{Tr}(\mathbb{1} X G_2 X^T)$ depends on X . Therefore, the regularization parameter γ plays a role when the number of nodes in G_1 and G_2 are different. Note that in solving relaxations or approximations of optimization (1.3) when X is no longer a permutation, γ can have an effect even for the same size graphs.

To illustrate the effect of the regularization parameter, consider example graphs G_1 and G_2 illustrated in Figure 1-b. Let X_1 and X_2 be mapping matrices that align nodes of G_1 to left and right subgraphs of G_2 , respectively. The number of matches and mismatches caused by X_1 are 8 and 7, respectively. The number of matches and mismatches caused by X_2 are 7 and 1, respectively. If we ignore the effect of mismatches (i.e., $\gamma = 0$ in (1.3)), X_1 leads to a larger graph alignment objective value compared to X_2 . However, if $\gamma > 1/6$, X_2 leads to a larger objective value compared to X_1 . Note that maximizing matches while ignoring mismatches favors parts of the larger graph with a higher edge density.

It is important to note that the notion of mismatches has been considered in other alignment frameworks as well. For example [10] considers aligning two images (modeled as graphs) knowing a pairwise similarity measure between nodes of the two graphs. Reference [10] uses mismatch terms (ignoring matches) to incorporate relational structure terms in the alignment optimization. Our generalized graph alignment optimization (1.3) does not require having a similarity matrix between nodes of the two graphs and uses both match and mismatch information to compute the alignment matrix.

The objective function of optimization (1.3) is not in the standard form of a quadratic assignment problem (QAP) since it has three terms. It is straightforward to show that the following optimization is an equivalent formulation:

$$\max_X \quad \text{Tr}((G_1 - \gamma \mathbb{1})X(G_2 - \gamma \mathbb{1})X^T). \tag{1.4}$$

This optimization is a standard QAP [14] which is computationally challenging to solve. In the next section we explain our algorithmic contributions to compute a solution for this optimization based on spectral decomposition of functions of adjacency matrices.

Reference [15] shows that approximating a solution of maximum quadratic assignment problem within a factor better than $2^{\log^{1-\epsilon} n}$ is in general not feasible in polynomial time. However, owing to numerous applications of QAP in different areas, several algorithms have been designed to solve it approximately. Some methods use exact search approaches based on branch-and-bound [16] and cutting plane [17]. These methods can only be applied to very small problem instances owing to their high computational complexity. Some methods attempt to solve the underlying QAP by linearizing the quadratic term and transforming the optimization into a mixed integer linear program (MILP) [18–21]. In practice the very large number of introduced variables and constraints in linearization of the QAP objective function poses an obstacle for solving the resulting MILP efficiently. Some methods use convex relaxations of the QAP to compute a bound on its optimal value [22–26]. The solutions provided by these methods may not be a feasible solution for the original quadratic assignment problem. Other methods to solve the graph alignment optimization include semidefinite [26, 27], non-convex [28], or Lagrangian [29–31] relaxations, Bayesian inference [32], message passing [33] or other heuristics [3, 4, 6, 34–38]. We will review these methods in Section 2. For more details about these methods, we refer readers to references [14, 39, 40]. In particular [40] provides a recent review of graph alignment methods by distinguishing between methods for deterministic and random graphs.

Spectral inference methods have received significant attention in problems such as graph clustering [41–45] where the underlying mixed integer program is tightly approximated with an optimization whose optimizers can be computed efficiently. However, the use of spectral techniques in the graph alignment problem has been limited [3, 4, 34, 35, 46, 47], partially owing to difficulty in connecting existing spectral graph alignment methods with relaxations of the underlying QAP. For example, [3] computes an alignment across biological networks using the top eigenvector of a graph which encodes neighborhood similarities. Reference [46] uses a spectral relaxation of QAP to compute a probabilistic subgraph matching when the number of nodes of graphs are the same, while [47] uses a heuristic multi-scale spectral signature of graphs to compute an alignment across them.

In this paper, we propose two spectral algorithms for solving the graph alignment optimization (1.4), namely *EigenAlign* (EA), and *LowRankAlign* (LRA):

1. **EigenAlign (EA)** computes the leading eigenvector of a function of adjacency matrices followed by a maximum weight bipartite matching optimization. EigenAlign can be applied to both directed and undirected graphs. We prove that for Erdős-Rényi graphs [48] and under some general conditions, EigenAlign is mean-field optimal ¹.

2. **LowRankAlign (LRA)** solves the graph alignment optimization by simultaneous alignment of eigenvectors of (transformations of) adjacency graphs, scaled by corresponding eigenvalues. LRA considers undirected graphs. LRA first solves the orthogonal relaxation of the underlying QAP using eigen decomposition of matrices. Then, it employs a rounding step as a projection in the direction of top eigenvectors of input matrices. We provide a bound on the performance of this projection step based on eigenvalues of input matrices and the orthogonal relaxation gap. Note that this rounding step is different than previously studied orthogonal projection, which has been shown to have a poor performance in practice [46].

¹Finding an isomorphic mapping across asymptotically large Erdős-Rényi graphs is a well studied problem and can be solved efficiently through canonical labeling [49]. Moreover Laszlo Babai has recently outlined his proof that the computational complexity of the general graph isomorphism problem is Quasipolynomial [50]. Note that in the graph alignment setup input graphs do not need to be isomorphic.

Through analytical performance characterization, simulations on several synthetic graphs, and real-data analysis, we show that our proposed graph alignment methods lead to improved performance compared to some existing graph alignment methods. Note that our proposed generalized graph alignment framework can also be adapted to some existing graph alignment packages. However, exploring this direction is beyond the scope of this article.

The rest of the paper is organized as follows. In Section 2, we review some existing graph alignment techniques and explain the relationship between graph alignment and graph isomorphism. In Section 3, we introduce the EigenAlign Algorithm and discuss its relationship with the underlying quadratic assignment problem. Moreover, we present the mean-field optimality of this method over random graphs, under some general conditions. In Section 4, we consider the trace formulation of the graph alignment optimization and introduce LowRankAlign. In Section 5, we compare performance of our method with some existing graph alignment methods over different synthetic graph structures. In Section 6, we use our graph alignment methods in comparative analysis of gene regulatory networks across different species.

2 Review of Prior Work

Graph alignment problem (1.4) is an example of a QAP [14]. In the following we briefly summarize previous works by categorizing them into four groups and explain advantages and shortcomings of each. For more details on these methods we refer readers to references [14, 39, 51].

1. Exact search methods: These methods provide a globally optimal solution for QAP. Examples of exact algorithms include methods based on branch-and-bound [16] and cutting plane [17]. Owing to their high computational complexity, they can only be applied to very small problem instances.

2. Linearizations: These methods attempt to solve QAP by eliminating the quadratic term in the objective function, transforming it into a mixed integer linear program (MILP). An existing MILP solver is applied to find a solution for the relaxed problem. Examples of these methods are Lawlers linearization [18], Kaufmann and Broeckx linearization [19], Frieze and Yadegar linearization [20], and Adams and Johnson linearization [21]. These linearizations can provide bounds on the optimal value of the underlying QAP [15]. Moreover [29–31] use Lagrangian relaxations to compute a solution for the QAP. In general, linearization of the QAP objective function is achieved by introducing many new variables and new linear constraints. In practice, the very large number of introduced variables and constraints poses an obstacle for solving the resulting MILP efficiently.

3. Semidefinite/convex relaxations: These methods aim to compute a bound on the optimal value of the graph alignment optimization by considering the alignment matrix in the intersection of orthogonal and stochastic matrices. The provided solution by these methods may not be a feasible solution for the original quadratic assignment problem. Examples of these methods include orthogonal relaxations [22], projected eigenvalue bounds [23], convex relaxations [24–26], and matrix splittings [27]. In particular, [27] introduces a convex relaxation of the underlying graph alignment optimization based on matrix splitting which provides bounds on the optimal value of the underlying QAP. The proposed semidefinite programming (SDP) method provides a bound on the optimal value and additional steps are required to derive a feasible solution. Moreover, owing to its computational complexity, it can only be used to align small graphs [27].

In the computer vision literature, [34, 35] use spectral techniques to solve QAP approximately by inferring a cluster of assignments over the feature graph. Then, they use a greedy approach

to reject assignments with low associations. Similarly, [46] uses a spectral relaxation of QAP to compute a probabilistic subgraph matching across images when the size of graphs are the same, while [47] uses a heuristic multi-scale spectral signature of graphs to compute an alignment across them.

4. Other methods: There are several other techniques to solve graph alignment optimization approximately. Some methods use Bayesian framework [32], or message passing [33], or some other heuristics [3, 4, 6]. In Section 5, we assess the performance of some of these graph alignment techniques through simulations.

Some graph alignment formulations aim to align paths [7] or subgraphs [8, 52, 53] across two (or multiple) graphs. The objective of these methods is different from the one of our graph alignment optimization where a bijective mapping across nodes of two graphs is desired according to a QAP. However solutions of these different methods may be related. For instance a bijective mapping across nodes of two graphs can provide information about conserved pathways and/or subgraphs across graphs, and vice versa.

The graph alignment formulation of (1.4) uses the structure of input graphs to find an alignment across their nodes. In practice, however, some other side information may be available such as node-node similarities. One way to incorporate such information in the formulation of (1.4) is to restrict the alignment across nodes of the two graphs whose similarities are greater than a threshold. This can be done by adding additional constraints to (1.4). We will explain this in more detail in Section 3.

2.1 Graph Alignment and Graph Isomorphism

The graph alignment optimization (1.3) is closely related to the problem of *graph isomorphism* defined as follows:

Definition 1 (Graph Isomorphism) *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two binary graphs. G_1 and G_2 are isomorphic if there exists a permutation matrix P such that $G_1 = PG_2P^T$.*

The computational problem of determining whether two finite graphs are isomorphic is called the *graph isomorphism problem*. Moreover given two isomorphic graphs G_1 and G_2 , in the graph isomorphism problem one aims to find the permutation matrix P such that $G_1 = PG_2P^T$. The computational complexity of this problem is unknown [54].

In the following lemma we formalize a connection between the graph alignment optimization and the classical graph isomorphism problem:

Lemma 1 *Let G_1 and G_2 be two isomorphic Erdős-Rényi graphs [48] such that $\Pr[G_1(i, j) = 1] = p$ and $G_2 = PG_1P^T$, where P is a permutation matrix. Let $p \neq 0, 1$. Then, for any selection of scores $s_1 > s_2 > s_3 > 0$, P maximizes the expected graph alignment objective function of Optimization (3.4). The expectation is over different realizations of G_1 and G_2 .*

Proof The proof is presented in Section 9.1. ■

The result of Lemma 1 can be extended to the case where edges of graphs are flipped through a random noise matrix:

Lemma 2 *Let G_1 be an Erdős-Rényi graph such that $\Pr[G_1(i, j) = 1] = p$. Let \tilde{G}_1 be a graph resulting from flipping edges of G_1 independently and randomly with probability q . Suppose $G_2 =$*

$P\tilde{G}_1P^T$ where P is a permutation matrix. Let $0 < p < 1/2$ and $0 \leq q < 1/2$. Then, for any selection of scores $s_1 > s_2 > s_3 > 0$, P maximizes the expected graph alignment objective function of Optimization (1.3). The expectation is over different realizations of G_1 and G_2 .

Proof The proof is presented in Section 9.1. ■

Finding an isomorphic mapping across sufficiently large Erdős-Rényi graphs can be done efficiently with high probability (w.h.p.) through canonical labeling [49]. Canonical labeling of a graph consists of assigning a unique label to each vertex such that labels are invariant under isomorphism. The graph isomorphism problem can then be solved efficiently by mappings nodes with the same canonical labels to each other [55]. One example of canonical labeling is the degree neighborhood of a vertex defined as a sorted list of neighborhood degrees of vertices [49]. Note that graph alignment formulation is more general than the one of graph isomorphism: graph alignment aims to find an optimal mappings across two graphs which are not necessarily isomorphic.

3 EigenAlign Algorithm

3.1 Problem Formulation and Notation

Let \mathbf{y} be a vectorized version of X . That is, \mathbf{y} is a vector of length n_1n_2 where, $y(i + (j' - 1)n_1) = X(i, j')$. To simplify notation, define $y_{i,j'} \triangleq X(i, j')$. Two mappings (i, j') and (r, s') can be matches which cause overlaps, can be mismatches which cause errors, or can be neutrals (Figure 1-a).

Definition 2 Suppose $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are undirected graphs. Let $\{i, r\} \subseteq V_1$ and $\{j', s'\} \subseteq V_2$ where $X(i, j') = 1$ and $X(r, s') = 1$. Then,

- (i, j') and (r, s') are matches if $(i, r) \in E_1$ and $(j', s') \in E_2$.
- (i, j') and (r, s') are mismatches if only one of the edges (i, r) and (j', s') exists.
- (i, j') and (r, s') are neutrals if none of the edges (i, r) and (j', s') exists.

Definition 2 can be extended to the case where G_1 and G_2 are directed graphs. In this case mappings (i, j') and (r, s') are matches/mismatches if they are matches/mismatches in one of the possible directions. However it is possible to have these mappings be matches in one direction while they are mismatches in the other direction (Figure 1-c). These mappings are denoted as *inconsistent mappings*, defined as follows:

Definition 3 Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two directed graphs and $\{i, r\} \subseteq V_1$ and $\{j', s'\} \subseteq V_2$ where $X(i, j') = 1$ and $X(r, s') = 1$. If edges $i \rightarrow r$, $r \rightarrow i$, and $j' \rightarrow s'$ exist, however, $s' \rightarrow j'$ does not exist, then mappings (i, j') and (r, s') are inconsistent.

Consider two undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. We form an *alignment graph* represented by adjacency matrix A in which nodes are mapping edges across the original graphs, and the edges capture whether the pair of mapping edges are matches, mismatches or neutrals (Figure 2).

Definition 4 Let $\{i, r\} \subseteq V_1$ and $\{j', s'\} \subseteq V_2$ where $X(i, j') = 1$ and $X(r, s') = 1$.

$$A[(i, j'), (r, s')] = \begin{cases} s_1, & \text{if } (i, j') \text{ and } (r, s') \text{ are matches,} \\ s_2, & \text{if } (i, j') \text{ and } (r, s') \text{ are neutrals,} \\ s_3, & \text{if } (i, j') \text{ and } (r, s') \text{ are mismatches,} \end{cases} \quad (3.1)$$

where s_1, s_2 , and s_3 are scores assigned to matches, neutrals, and mismatches, respectively. Without loss of generality we assume $s_1 > s_2 > s_3 > 0$.

We can re-write (3.1) as follows:

$$A[(i, j'), (r, s')] = (s_1 + s_2 - 2s_3)G_1(i, r)G_2(j', s') + (s_3 - s_2)(G_1(i, r) + G_2(j', s')) + s_2. \quad (3.2)$$

We can summarize (3.1) and (3.2) as follows:

$$A = (s_1 + s_2 - 2s_3)(G_1 \otimes G_2) + (s_3 - s_2)(G_1 \otimes \mathbb{1}_{n_2}) + (s_3 - s_2)(\mathbb{1}_{n_1} \otimes G_2) + s_2(\mathbb{1}_{n_1} \otimes \mathbb{1}_{n_2}), \quad (3.3)$$

where \otimes represents matrix Kronecker product, and $\mathbb{1}_n$ is an $n \times n$ matrix whose elements are all ones.

A similar scoring scheme can be used for directed graphs. When graphs are directed, some mappings can be inconsistent according to Definition 3, i.e., they are matches in one direction and mismatches in another. Scores of inconsistent mappings can be assigned randomly to matched/mismatched scores, or to an average score of matches and mismatches (i.e., $(s_1 + s_3)/2$). For random graphs, inconsistent mappings are rare events. For example, suppose graph edges are distributed according to a Bernoulli distribution with parameter p . Then, the probability of having an inconsistent mapping for a particular pair of paired nodes across graphs is equal to $4p^3(1-p)$. Therefore, their effect in graph alignment is negligible, particularly for large sparse graphs. Throughout the paper, for directed graphs we assume inconsistent mappings have negligible effect, unless we mention the importance of such inconsistency explicitly.

In practice some mapping edges across two graphs may not be possible, owing to additional side information. The set of possible mapping edges across two graphs is denoted by $\mathcal{R} = \{(i, j') : i \in V_1, j' \in V_2\}$. If $\mathcal{R} = V_1 \times V_2$, the problem of graph alignment is called *unrestricted*. If some mappings across two graphs are prevented (i.e., $X(i, j') = y_{i, j'} = 0$, for $(i, j') \notin \mathcal{R}$), then the problem of graph alignment is called *restricted*.

Using the vectorized version of X , the graph alignment optimization (1.2) can be written as follows:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T A \mathbf{y}, \\ & \sum_i y_{i, j'} \leq 1, \quad \forall i \in V_1, \\ & \sum_{j'} y_{i, j'} \leq 1, \quad \forall j' \in V_2, \\ & y_{i, j'} \in \{0, 1\}, \quad \forall (i, j') \in V_1 \times V_2, \\ & y_{i, j'} = 0, \quad \forall (i, j') \notin \mathcal{R}, \end{aligned} \quad (3.4)$$

where A is defined according to (3.2) and $\mathcal{R} \subseteq V_1 \times V_2$ is the set of possible mapping edges across two graphs.

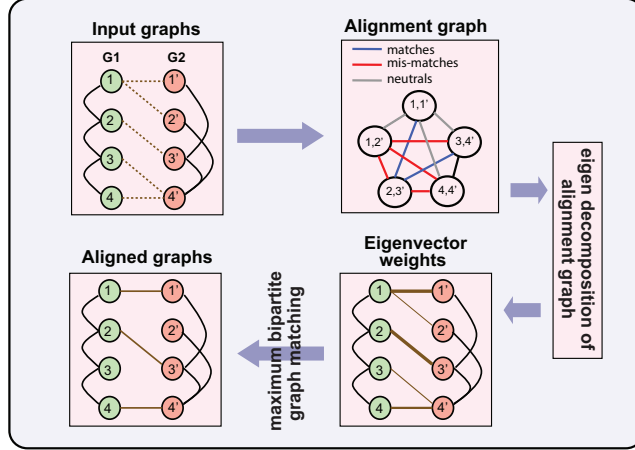


Figure 2: The Framework of EigenAlign algorithm 1.

3.2 EigenAlign Algorithm

We now introduce *EigenAlign* (EA) algorithm which computes a solution for the graph alignment optimization (3.4) leveraging spectral properties of graphs:

Algorithm 1 (EigenAlign Algorithm) Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two binary graphs whose corresponding alignment graph is denoted by A according to (3.2). *EigenAlign* algorithm solves the graph alignment optimization (3.4) in two steps:

Step 1, An Eigenvector Computation Step: In this step we compute \mathbf{v} , an eigenvector of the alignment graph A with the maximum eigenvalue.

Step 2, A Linear Assignment Step: In this step we solve the following maximum weight bipartite matching optimization:

$$\begin{aligned}
 \max_{\mathbf{y}} \quad & \mathbf{v}^T \mathbf{y}, \\
 \sum_{j'} y_{i,j'} & \leq 1, \quad \forall i \in V_1, \\
 \sum_i y_{i,j'} & \leq 1, \quad \forall j' \in V_2, \\
 y_{i,j'} & \in \{0, 1\}, \quad \forall (i, j') \in V_1 \times V_2, \\
 y_{i,j'} & = 0, \quad \forall (i, j') \notin \mathcal{R}.
 \end{aligned} \tag{3.5}$$

Algorithm 1 can be extended to directed graphs G_1^{dir} and G_2^{dir} as well. This framework is depicted in Figure 2. Below we provide intuition on different steps of the EigenAlign algorithm. For simplicity we assume all mappings across graphs are possible (i.e., $\mathcal{R} = \{(i, j') : \forall i \in V_1, \forall j' \in V_2\}$). In the restricted graph alignment setup, without loss of generality, one can eliminate rows and columns of the alignment matrix corresponding to mappings that are not allowed.

In the eigen decomposition step of EigenAlign, we ignore bijective constraints (i.e., constraints $\sum_i y_{i,j'} \leq 1$ and $\sum_{j'} y_{i,j'} \leq 1$) because they will be satisfied in the second step of the algorithm through a linear optimization. Under these assumptions Optimization (3.4) can be simplified to

the following optimization:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T \mathbf{A} \mathbf{y}, \\ & y_{i,j'} \in \{0, 1\}, \quad \forall (i, j') \in V_1 \times V_2. \end{aligned} \tag{3.6}$$

To approximate a solution of this optimization, we replace integer constraints with constraints over a hyper-sphere restricted by hyper-planes (i.e., $\|\mathbf{y}\|_2 \leq 1$ and $\mathbf{y} \geq 0$). Thus, optimization (3.6) is simplified to the following:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T \mathbf{A} \mathbf{y}, \\ & \|\mathbf{y}\|_2 \leq 1, \\ & \mathbf{y} \geq 0. \end{aligned} \tag{3.7}$$

In the following, we show that \mathbf{v} , the leading eigenvector of the alignment matrix A , is an optimal solution of Optimization (3.7). Suppose \mathbf{y}_1 is an optimal solution of Optimization (3.7). Let \mathbf{y}_2 be a solution of the following optimization without non-negativity constraints:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T \mathbf{A} \mathbf{y}, \\ & \|\mathbf{y}\|_2 \leq 1. \end{aligned} \tag{3.8}$$

Following the Rayleigh-Ritz formula [56], the leading eigenvector of the alignment matrix is an optimal solution of Optimization (3.8) (i.e., $\mathbf{y}_2 = \mathbf{v}$). Now we use the following theorem to show that in fact $\mathbf{y}_1 = \mathbf{v}$:

Theorem 1 *Suppose A is a matrix whose elements are strictly positive. Let \mathbf{v} be an eigenvector of A corresponding to the largest eigenvalue. Then, $\forall i, v_i > 0$. Moreover, all other eigenvectors must have at least one negative, or non-real component.*

Proof See e.g., reference [57] (Theorem 1). ■

Since \mathbf{y}_2 is a solution of Optimization (3.8), we have $\mathbf{y}_2^T \mathbf{A} \mathbf{y}_2 \geq \mathbf{y}_1^T \mathbf{A} \mathbf{y}_1$. Using this inequality along with the Perron-Frobenius Theorem lead to $\mathbf{y}_1 = \mathbf{v}$, as the unique solution of optimization (3.7).

The solution of the eigen decomposition step assigns weights to all possible mapping edges across graphs ignoring bijective constraints (constraints $\sum_{j'} y_{i,j'} \leq 1$ and $\sum_i y_{i,j'} \leq 1$). However, in the graph alignment setup, each node in one graph can be mapped to at most one node in the other graph. To satisfy these constraints, we use eigenvector weights in a linear optimization framework of maximum weight bipartite matching setup of Optimization (3.5) [58].

3.3 Computational Complexity of EigenAlign

Let the number of nodes of graphs G_1 and G_2 be $\mathcal{O}(n)$. Let $k = |\mathcal{R}|$ be the number of possible mappings across two graphs. In an unrestricted graph alignment setup, we have $k = \mathcal{O}(n^2)$. However, in a restricted graph alignment, k may be significantly smaller than n^2 . EigenAlign has three steps:

- (i) Forming an alignment graph A that has a computational complexity of $\mathcal{O}(k^2)$, as all pairs of possible mappings should be considered.

(ii) An eigen decomposition step where we compute the leading eigenvector of the alignment graph. This operation can be performed in $\mathcal{O}(k^2)$ computational complexity using QR algorithms and/or power methods [59]. Therefore, the worst case computational complexity of this part is $\mathcal{O}(k^2)$.

(iii) A maximum weight bipartite matching algorithm step, that can be solved efficiently using linear programming or the Hungarian algorithm [58]. The worst case computational complexity of this step is $\mathcal{O}(n^3)$. If the set \mathcal{R} has a specific structure (e.g., small subsets of nodes in one graph are allowed to be mapped to small subsets of nodes in the other graph), this cost can be reduced significantly.

Proposition 1 *The worst case computational complexity of the EigenAlign Algorithm is $\mathcal{O}(k^2 + n^3)$.*

Remark 1 For large graphs, to reduce the overall computational complexity, the linear assignment optimization may be replaced by a greedy bipartite matching algorithm (e.g., [60]). In the greedy matching approach, at each step, the heaviest possible mapping is added to the current matching until no further mappings can be added. It is straightforward to show that this greedy algorithm finds a bipartite matching whose weight is at least half the optimum. The computational complexity of this greedy algorithm is $\mathcal{O}(k \log(k) + nk)$.

If we only consider matches in the graph alignment optimization (i.e., $s_2 = s_3 = 0$ in (1.2)), the complexity of the eigen decomposition step can be reduced, since we need to compute top eigenvectors of sparse adjacency matrices. By considering mismatches, eigenvector computation should be performed over dense matrices, which require a higher computational complexity.

3.4 Mean-field Optimality of EigenAlign Over Erdős-Rényi Graphs

Here we analyze the performance of the EigenAlign algorithm over Erdős-Rényi graphs, for both isomorphic and non-isomorphic cases, under two different noise models. While real graphs often have different structures than Erdős-Rényi graphs, we consider this family of graphs in this section owing to their analytical tractability.

Suppose $G_1 = (V_1, E_1)$ is an undirected Erdős-Rényi graph with n nodes where $Pr[G_1(i, j) = 1] = p$ for $1 \leq i, j \leq n$. Suppose \tilde{G} is a noisy version of the graph G_1 . We consider two different noise models in this section:

Noise Model I: In this model we have,

$$\tilde{G}_1 \triangleq G_1 \odot (\mathbb{1} - Q) + (\mathbb{1} - G_1) \odot Q, \quad (3.9)$$

where \odot represents the Hadamard product, $\mathbb{1}$ is the matrix of all ones, and Q is a binary symmetric random matrix whose edges are drawn i.i.d. from a Bernoulli distribution with $Pr[Q(i, j) = 1] = p_e$. In words, the operation $G_1 \odot (\mathbb{1} - Q) + (\mathbb{1} - G_1) \odot Q$ flips edges of G_1 uniformly randomly with probability p_e .

Noise Model II: In this model we have,

$$\tilde{G}_1 \triangleq G_1 \odot (\mathbb{1} - Q) + (\mathbb{1} - G_1) \odot Q', \quad (3.10)$$

where Q and Q' are binary symmetric random matrices whose edges are drawn i.i.d. from a Bernoulli distribution with $Pr[Q(i, j) = 1] = p_e$ and $Pr[Q'(i, j) = 1] = p_{e_2}$. Under this model,

edges of G_1 flip uniformly randomly with probability p_e , while non-connecting tuples in G_1 will be connected in \tilde{G}_1 with probability p_{e_2} . Because G_1 is an Erdős-Rényi graph with parameter p , choosing

$$p_{e_2} = \frac{pp_e}{1-p}, \quad (3.11)$$

leads to having the expected density of graphs G_1 and G_2 be equal to p .

Using either model I (3.9) or model II (3.10) for \tilde{G}_1 , we define G_2 as follows:

$$G_2 \triangleq P\tilde{G}_1P^T, \quad (3.12)$$

where P is a permutation matrix. Recall that \mathcal{R} is the set of possible mapping edges across graphs G_1 and G_2 . Throughout this section, we assume that we are in the restricted graph alignment regime where $|\mathcal{R}| = kn$ for $k > 1$. The n true mapping edges ($i \leftrightarrow i'$ if $P = I$) are included in \mathcal{R} , while the remaining $(k-1)n$ mappings are selected uniformly randomly.

Let S_{true} be the set of true mapping edges between G_1 and G_2 , i.e., $S_{\text{true}} \triangleq \{(i, j) : P(i, j) = 1\}$. We define $S_{\text{false}} = \mathcal{R} - S_{\text{true}}$ as the set of incorrect mapping edges between the two graphs. Moreover, we choose scores assigned to matches, neutrals and mismatches as $s_1 = \alpha + \epsilon$, $s_2 = 1 + \epsilon$ and $s_3 = \epsilon$, respectively, where $\alpha > 1$ and $0 < \epsilon \ll 1$. These selections satisfy score conditions $s_1 > s_2 > s_3 > 0$ and lead to the regularization parameter $\gamma = 1/(1 + \alpha)$ in (1.3).

Theorem 2 *Let A be the alignment graph between G_1 and G_2 as defined in (3.2) with $s_1 = \alpha + \epsilon$, $s_2 = 1 + \epsilon$ and $s_3 = \epsilon$. Let \mathbf{v} be the eigenvector of $\mathbb{E}[A]$ corresponding to the largest eigenvalue, where the expectation is over realizations of G_1 , G_2 and \mathcal{R} . Then, under both noise models (3.9) and (3.10), if $0 < p < 1/2$, and $0 \leq p_e < 1/2$, as $n \rightarrow \infty$,*

$$\mathbf{v}(t_1) > \mathbf{v}(t_2), \quad \forall t_1 \in S_{\text{true}} \text{ and } \forall t_2 \in S_{\text{false}}.$$

In noise models (3.9) and (3.10), if we put $p_e = 0$, then G_2 is isomorphic with G_1 because there exists a permutation matrix P such that $G_2 = PG_1P^T$. For this case, we have the following Corollary:

Corollary 1 *Let G_1 and G_2 be two isomorphic Erdős-Rényi graphs with n nodes such that $G_1 = PG_2P^T$, where P is a permutation matrix. Under the conditions of Theorem 2, as $n \rightarrow \infty$, $\mathbf{v}(t_1) > \mathbf{v}(t_2)$ where where \mathbf{v} is the top eigenvector of the expected alignment graph, t_1 is a true mapping edge and t_2 is a false mapping edge between the two graphs.*

We present proofs of Theorem 2 and Corollary 1 in Sections 9.2 and 9.3.

In the EigenAlign algorithm, we use values of the top eigenvector of the alignment graph in a maximum weight bipartite matching optimization to extract bijective mappings between the two graphs. Thus, if true mapping edges obtained higher eigenvector scores compared to the false one, the EigenAlign algorithm would infer optimal mappings between the two graphs. Theorem 2 indicates that, in an expectation sense, true mapping edges obtain larger eigenvector scores compared to the false ones when $|\mathcal{R}| = kn$. In Section 5 and through simulations, we show that the error of the EigenAlign algorithm is empirically small even in an unrestricted graph alignment setup.

4 LowRankAlign Algorithm

In this section, we introduce a graph alignment algorithm that uses higher-order eigenvectors of (transformations of) adjacency graphs to align their structures. We refer to this extension as *LowRankAlign* (LRA). LRA can be useful specially in cases where leading eigenvectors of graphs are not informative. This case occurs for instance in the alignment of regular graph structures. Moreover, LRA does not require an explicit formation of the alignment graph which can be costly for large graphs if all mappings across graphs are possible.

Higher order eigenvectors have been used in other spectral inference problems such as graph clustering [41–45] and the matrix coupling [61, 62]. Moreover reference [63] has used higher order eigenvectors of the graph Laplacian to embed large graphs on a low-dimensional isometric space to compute an inexact matching. Our goal in this section is to provide a principled framework to exploit higher order eigenvectors in the graph alignment problem.

Here we assume graphs are symmetric. For simplicity we assume $n_1 = n_2 = n$. All discussions can be extended to the case where $n_1 \neq n_2$. Moreover, to simplify analysis, we assume singular values of matrices have multiplicity of one. Let \square be the set of all permutation matrices of size $n \times n$. Thus, the graph alignment optimization can be written as follows ²:

$$\max_{X \in \square} \text{Tr}(G_1 X G_2 X^T), \quad (4.1)$$

Let X^* be an optimal solution of optimization (4.1). Finding an optimal solution of this optimization is known to be NP-hard [15]. If $X \in \square$, we have

$$\text{Tr}(G_1 X G_2 X^T) = \text{Tr}((G_1 + \delta_1 I) X (G_2 + \delta_2 I) X^T) + \text{constant}. \quad (4.2)$$

In other words we can add and subtract multiples of identity to make the resulting symmetric matrices positive definite, without changing the structure of the problem. Thus, without loss of generality, we assume that matrices G_1 and G_2 are positive semi-definite.

We compute a solution for Optimization (4.1) in two steps:

(i) **The Relaxation Step:** First, we compute a solution X_0 to a relaxation of Optimization (4.1) over orthogonal matrices. Other relaxations can be considered as well. X_0 may not be a valid permutation matrix.

(ii) **The Rounding Step:** We propose a rounding step using projection in the direction of eigenvectors of (transformations of) adjacency graphs scaled by their corresponding eigenvalues.

Below we explain these steps with more details:

The Relaxation step: Let Γ be a set that contains all permutation matrices (i.e., $\square \subseteq \Gamma$). An example of Γ is the set of orthogonal matrices. Let X_0 be a solution of the following optimization:

$$\max_{X \in \Gamma} \text{Tr}(G_1 X G_2 X^T), \quad (4.3)$$

If Γ is assumed to be the set of orthogonal matrices (i.e., $\Gamma = \mathcal{O}$), an optimal solution of optimization (4.3) can be found using eigen decomposition of matrices G_1 and G_2 as follows:

²To consider the generalized graph alignment formulation of (1.4), one can replace G_1 and G_2 with $G_1 - \gamma \mathbb{1}$ and $G_2 - \gamma \mathbb{1}$ in (4.1), respectively.

Theorem 3 Suppose v_i and u_i are eigenvectors of symmetric matrices G_1 and G_2 , respectively. Let V and U be eigenvector matrices whose i -th columns are v_i and u_i , respectively. Then,

$$X_0 = VU^T = \sum_{i=1}^n v_i u_i^T, \quad (4.4)$$

is an optimal solution of optimization (4.3) over orthogonal matrices (i.e., $\Gamma = \mathcal{O}$).

Proof See Section 6.1 of reference [22]. ■

Theorem 3 characterizes an optimal solution of the orthogonal relaxation of the graph alignment optimization. A similar argument can be constructed for eigenvectors of the matrix G_2 . Let

$$\mathcal{X}_0 \triangleq \left\{ X_0 : X_0 = \sum_{i=1}^n s_i v_i u_i^T, \mathbf{s} \in \{-1, 1\}^n \right\}, \quad (4.5)$$

where s_i is the i -th component of the vector \mathbf{s} . The set \mathcal{X}_0 represents multiple optimal solutions of optimization (4.3) when $\Gamma = \mathcal{O}$. It is because if \mathbf{v} is an eigenvector of a matrix corresponding to the eigenvalue λ , $-\mathbf{v}$ is also an eigenvector of the same matrix with the same eigenvalue. \mathcal{X}_0 can have at most 2^n distinct members.

The Rounding step: X_0 may not be a valid permutation matrix. One way to find a permutation matrix using X_0 is to project X_0 over the space of permutation matrices Π :

$$\begin{aligned} \max \quad & Tr(XX_0^T), \\ & X \in \Pi. \end{aligned} \quad (4.6)$$

However, it has been shown that an optimal solution of optimization (4.6) has a poor performance in practice [64]. In the following, we propose an alternative algorithm to compute a permutation matrix using X_0 with a certain performance guarantee. Consider the following optimization:

$$\begin{aligned} \max \quad & Tr(G_1 X_0 G_2 X^T), \\ & X \in \Pi, \\ & X_0 \in \mathcal{X}_0. \end{aligned} \quad (4.7)$$

For a fixed X_0 , this is a maximum weight bipartite matching optimization which can be solved exactly using linear programming. Let X_{lin}^* be an optimal solution of optimization (4.7). Define

$$\begin{aligned} f(X) &\triangleq Tr(G_1 X G_2 X^T), \\ \tilde{f}(X) &\triangleq Tr(G_1 X_0 G_2 X_0^T) + 2Tr(G_1 X_0 G_2 (X - X_0)^T). \end{aligned} \quad (4.8)$$

Theorem 4 Let X^* and X_{lin}^* be optimal solutions of optimizations (4.1) and (4.7), respectively. We have,

$$|f(X^*) - \tilde{f}(X_{lin}^*)| \leq \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2), \quad (4.9)$$

where $\sigma_i(G_a)$ represents the i -th largest singular value of matrix G_a , for $a = 1, 2$, and ϵ is a bound on the relaxation gap (i.e., $\min_{X_0 \in \mathcal{X}_0} \|X^* - X_0\|_{op} \leq \epsilon$). Note that $\|\cdot\|_{op}$ indicates the matrix operator norm.

Proof See Section 9.4. ■

Optimization (4.7) can be simplified to the following optimization which finds a valid permutation matrix using the orthogonal relaxation of the graph alignment optimization:

$$\begin{aligned} \max \quad & Tr\left(\left(\sum_{i=1}^n \lambda_i(G_1)\lambda_i(G_2)s_i v_i u_i^T\right)X^T\right), \\ & X \in \Pi, \\ & \mathbf{s} \in \{-1, 1\}^n, \end{aligned} \tag{4.10}$$

where $\lambda_i(G_a)$ is the i -th largest eigenvalue of G_a for $a = 1, 2$. The objective function of optimization (4.10) simplifies the graph alignment problem to the simultaneous alignment of eigenvectors whose contributions in the overall alignment score are weighed by their corresponding eigenvalues. However, there are possibly exponentially many optimal solutions for optimization (4.10) and obtaining their resulting permutation matrices would be computationally infeasible. Because contributions of eigenvectors with small eigenvalues to the objective function of optimization (4.10) are small, one can instead as a heuristic, presumably solve the following optimization based on the low rank approximation of the objective function:

Algorithm 2 (LowRankAlign Algorithm) *The following optimization summarizes the LRA algorithm:*

$$\begin{aligned} \max \quad & Tr\left(\left(\sum_{i=1}^k s_i \lambda_i(G_1)\lambda_i(G_2)v_i u_i^T\right)X^T\right), \\ & X \in \Pi, \\ & s_i \in \{-1, 1\}, \quad \forall 1 \leq i \leq k. \end{aligned} \tag{4.11}$$

where k is a constant that determines the rank of the affinity matrix.

In the restricted graph alignment setup, some mapping edges across two graphs may not be allowed. In that case, one can set the affinity weights (i.e., weights used in the maximum weight bipartite matching step) of such pairs in optimization (4.11) to be $-\infty$.

5 Performance Evaluation Over Synthetic Graphs

Here we compare the performance of the proposed graph alignment algorithms (LRA and EA) against some other graph alignment methods including Natalie2 [30,31], GHOST [47], IsoRank [3], NetAlign [33], Klau’s approach [29] as well as an SDP-based method [27] through simulations. Natalie2 and Klau’s approach use Lagrange multipliers to relax the underlying quadratic assignment problem. IsoRank is a global graph alignment method that uses an iterative approach to align nodes across two graphs based on their neighborhood similarities, while GHOST uses a heuristic multi-scale spectral signature of graphs to compute an alignment across them. NetAlign formulates the alignment problem in a quadratic optimization framework and uses message passing to approximately solve it. The SDP-based method [27] uses a convex relaxation of the underlying QAP based on matrix splitting. In our simulations, we use default parameters of these methods.

We report the performance of proposed EigenAlign (EA) and LowRankAlign (LRA) Algorithms for $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5 - \epsilon\}$ where $\epsilon = 0.001$. In general, this parameter can be tuned in

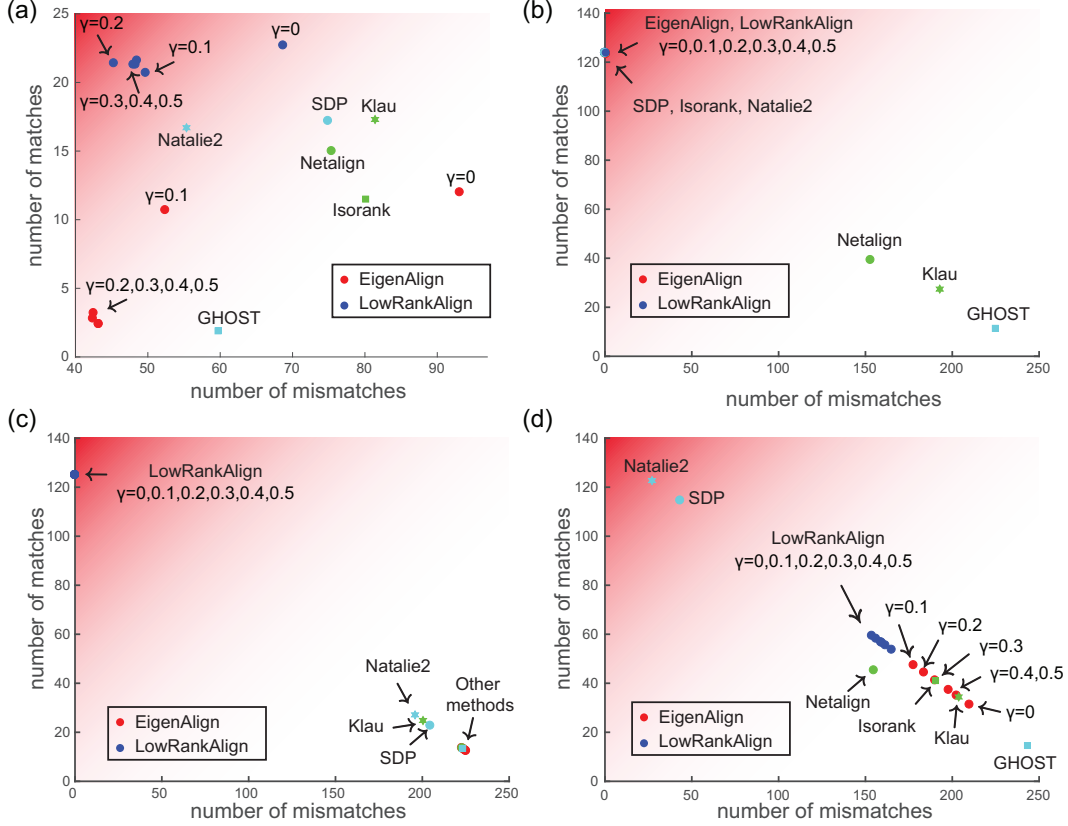


Figure 3: Performance evaluation of different graph alignment methods over (a) stochastic block models, (b) isomorphic Erdős-Rényi graphs, (c) isomorphic random regular graphs, and (d) noisy power law graphs. Experiments have been repeated 10 times in each case. For each method the average number of matches and mismatches have been shown. The high-match low-mismatch area has been highlighted by red shades.

different applications using standard machine learning techniques such as cross validations [65]. For LRA we use top $k = 3$ eigenvectors of input graphs as larger values of k did not have a significant effect on the results. We consider four different setups:

- G_1 is an Erdős-Rényi graph with $n_1 = 25$ nodes and the density parameter 0.1. G_2 is a stochastic block model with two blocks each with 25 nodes (i.e., $n_2 = 50$). Edge densities within blocks are 0.1 and 0.3, and the edge density across blocks is 0.05.
- G_1 and G_2 are isomorphic Erdős-Rényi graphs with $n_1 = n_2 = 50$ nodes with an edge density 0.1.
- G_1 and G_2 are isomorphic random regular graphs with $n_1 = n_2 = 50$ nodes whose edge density parameters are 0.1.
- G_1 is a power law graph [66] constructed as follows: we start with a random subgraph with 5 nodes. At each iteration, a node is added to the graph connecting to three existing nodes

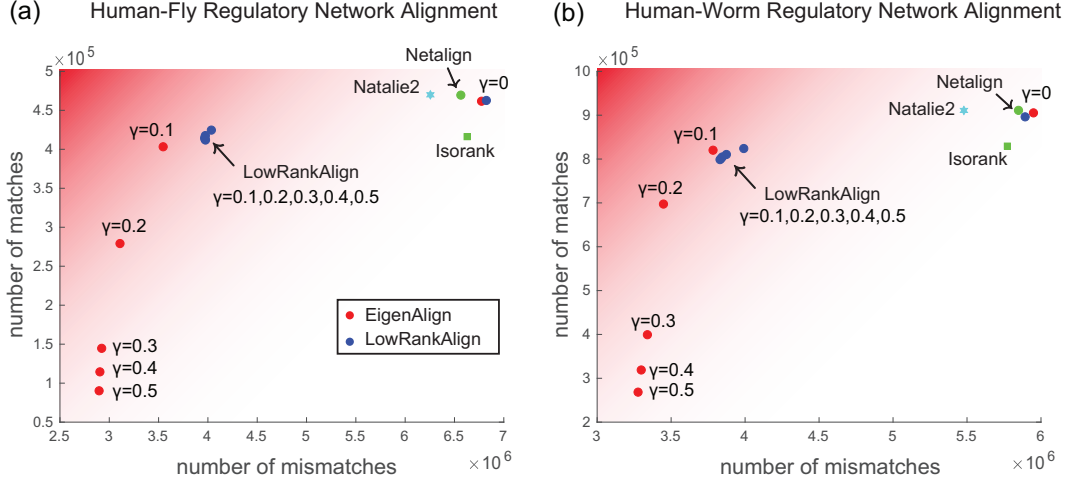


Figure 4: Performance evaluation of different graph alignment methods over (a) human-fly and (b) human-worm gene regulatory graphs. The high-match low-mismatch area has been highlighted by red shades.

with probabilities proportional to their degrees. This process is repeated till the number of nodes in the graph is equal to $n_1 = 50$. Then we construct G_2 according to the noise model (3.10) with $p_e = 0.05$. In (3.10) we use the density of G_1 as parameter p .

Figure 3 shows the number of matches and mismatches caused by different graph alignment methods in four considered setups for an unrestricted graph alignment problem. In the stochastic block model case (panel a) LRA outperforms other methods in terms of resulting in large number of matches and few mismatches. Since LRA with $\gamma = 0$ ignores the effect of mismatches, it results in a slightly larger number of matches compared to the case with $\gamma \neq 0$. At the same time LRA with $\gamma = 0$ results in a larger number of mismatches compared to the case with $\gamma \neq 0$. This highlights the effect of considering mismatches in the generalized graph alignment formulation (1.3) when graphs have different sizes and heterogenous edge densities.

Over isomorphic Erdős-Rényi graphs (panel b), EA, LRA, Isorank, SDP and Natalie2 have the best performance of achieving the highest number of matches and zero mismatches. Netalign, Klau and GHOST have poor performances in this case. Note that some of these methods are designed for very sparse graphs and for the restricted graph alignment setup. This may partially explain the poor performance of these methods.

Over isomorphic random graphs (panel c) LRA outperforms other methods achieving the highest number of matches and zero mismatches. The performance of LRA is also robust against parameter γ . Note that the alignment of regular graph structures is one of the most difficult graph alignment cases because of homogeneity of node degrees. The fact that LRA performs well in this case while all other methods have poor performance illustrates the effectiveness of using higher order eigenvectors in aligning homogenous graph structures. Finally, over noisy power law graphs (panel d) Natalie2 and SDP outperform other methods. The performance of LRA in this case is higher than other methods except Natalie2 and SDP.

6 Performance Evaluation Over Gene Networks

Here we apply graph alignment methods to compare gene regulatory graphs across human, fly and worm species. Comparative graph analysis in evolutionary studies often requires having a one-to-one mapping across genes of two or multiple networks. However, since human, fly and worm are distant species and as a result, many gene families have undergone extensive duplications and losses, we observe non-bijective homolog mappings across their genes [67]. For example, one gene in human can be homologous to multiple genes in fly and vice versa. To infer bijective mappings as a subset of homolog genes across species, we use graph alignment methods. We use regulatory networks that are inferred by integrating genome-wide functional and physical genomics datasets from ENCODE and modENCODE consortia (see the Appendix for more details).

Similarly to our discussion in Section 5 we report the performance of proposed EigenAlign (EA) and LowRankAlign (LRA) methods for $\gamma \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5 - \epsilon\}$ where $\epsilon = 0.001$. For LRA we use top $k = 2$ eigenvectors of input graphs. We also assess the performance of NetAlign, IsoRank, and Natalie2 in our real data analysis. We exclude Klau’s approach [29] and the SDP-based method of [27] from our analysis in this section owing to their high memory and computational complexity. Moreover the GHOST method failed to run over these networks owing to some implementation errors.

Figure 4 shows the number of matches and mismatches caused by different graph alignment methods across human-fly and human-worm networks. In both cases EA and LRA with $\gamma = 0$ (i.e., ignoring mismatches) have a comparable performance to other methods. However, by changing γ we observe a trade-off between number of caused matches and mismatches. For example, in the human-fly network alignment case LRA with a non-zero γ results in approximately 2-fold decrease in the number of mismatches while the number of caused matches decreases by approximately 10%. This highlights the effect of considering mismatches in the graph alignment optimization. To substantiate these inferences, further experiments should be performed to determine the involvement of inferred conserved gene interactions in different biological processes, which is beyond the scope of the present paper.

7 Conclusion

In this paper, we made two main contributions to the field of graph alignment. Firstly, we proposed a generalized graph alignment formulation that considers both matches and mismatches in a standard QAP formulation. We showed that this can be critical in applications where graphs have different sizes and heterogenous edge densities. Secondly, we proposed two graph alignment algorithms which employ spectral decompositions of functions of adjacency graphs followed by a maximum weight bipartite matching optimization. One of our proposed methods simplifies the graph alignment optimization to simultaneous alignment of eigenvectors of (transformations of) adjacency graphs scaled by corresponding eigenvalues. We demonstrated effectiveness of the proposed methods theoretically for certain classes of graphs and over various synthetic and real graph models.

8 Code

We provide code for the proposed method in the following link: <https://github.com/SoheilFeizi/spectral-graph-alignment>

9 Proofs

In this section, we present proofs of the main results of the paper.

9.1 Proofs of Lemmas 1 and 2

First we prove Lemma 1. Let A be the alignment graph of G_1 and G_2 . By a slight abuse of notation, we use A as the adjacency matrix of the alignment graph as well. Suppose \tilde{P} is a permutation matrix where $\rho \triangleq \frac{1}{2n} \|P - \tilde{P}\| > 0$. Let \mathbf{y} and $\tilde{\mathbf{y}}$ be vectorized versions of permutation matrices P and \tilde{P} , respectively. Then, we have,

$$\begin{aligned}
\frac{1}{n^2} \mathbb{E}[\tilde{\mathbf{y}}_2^T A \tilde{\mathbf{y}}_2] &= (1 - \rho)[ps_1 + (1 - p)s_2] \\
&+ \rho[p^2s_1 + (1 - p)^2s_2 + 2p(1 - p)s_3] \\
&< (1 - \rho)[ps_1 + (1 - p)s_2] + \rho[p^2s_1 + (1 - p)^2s_2 + p(1 - p)(s_1 + s_2)] \\
&= (1 - \rho)[ps_1 + (1 - p)s_2] + \rho[ps_1 + (1 - p)s_2] \\
&= ps_1 + (1 - p)s_2 \\
&= \frac{1}{n^2} \mathbb{E}[\mathbf{y}_1^T A \mathbf{y}_1].
\end{aligned} \tag{9.1}$$

Now we prove Lemma 2: Similarly to the proof of Lemma 1, let A be the alignment graph of G_1 and G_2 and suppose \tilde{P} is a permutation matrix where $\rho \triangleq \frac{1}{2n} \|P - \tilde{P}\| > 0$. Let \mathbf{y} and $\tilde{\mathbf{y}}$ be vectorized versions of permutation matrices P and \tilde{P} , respectively. Define a' and b' as follows:

$$\begin{aligned}
a' &\triangleq p(1 - q)s_1 + (1 - p)(1 - q)s_2 + (pq + (1 - p)q)s_3, \\
b' &\triangleq (p^2(1 - q) + pq(1 - p))s_1 \\
&\quad + ((1 - p)^2(1 - q) + pq(1 - p))s_2 \\
&\quad + (2p(1 - p)(1 - q) + 2p^2q)s_3.
\end{aligned} \tag{9.2}$$

Thus,

$$a' - b' = p(1 - p)(1 - 2q)(s_1 + s_2 - 2s_3) + q(1 - 2p)s_3. \tag{9.3}$$

Because $s_1 > s_2 > s_3$, we have, $s_1 + s_2 - 2s_3 > 0$. Because $0 < p < 1/2$ and $0 \leq q < 1/2$, we have $(1 - 2p) > 0$ and $(1 - 2q) > 0$. Therefore, according to (9.3), $a' > b'$. Thus we have,

$$\frac{1}{n^2} \mathbb{E}[\tilde{\mathbf{y}}^T A \tilde{\mathbf{y}}] = (1 - \rho)a' + \rho b' < a' = \frac{1}{n^2} \mathbb{E}[\mathbf{y}^T A \mathbf{y}].$$

9.2 Proof Of Corollary 1

Without loss of generality and to simplify notations, we assume the permutation matrix P is equal to the identity matrix I , i.e., the isomorphic mapping across G_1 and G_2 is $\{1 \leftrightarrow 1', 2 \leftrightarrow 2', \dots, n \leftrightarrow n'\}$ (otherwise, one can relabel nodes in either G_1 or G_2 to have P equal to the identity matrix). Therefore, $G_1(i, j) = G_2(i', j')$ for all $1 \leq i, j \leq n$. Recall that \mathbf{y} is a vector of length kn which has weights for all possible mapping edges $(i, j') \in \mathcal{R}$. To simplify notations and without loss of generality, we re-order indices of vector \mathbf{y} as follows:

- The first n indices of \mathbf{y} correspond to correct mappings, i.e., $y(1) = y_{1,1'}, y(2) = y_{2,2'}, \dots, y(n) = y_{n,n'}$.
- The remaining $(k-1)n$ indices of \mathbf{y} correspond to incorrect mappings. e.g., $y(n+1) = y_{1,2'}, y(n+2) = y_{1,3'}, \dots, y(kn) = y_{r,s'}$ ($r \neq s$).

Therefore, we can write,

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix},$$

where \mathbf{y}_1 and \mathbf{y}_2 are vectors of length n and $(k-1)n$, respectively.

We re-order rows and columns of the alignment matrix A accordingly. Define the following notations: $\mathcal{S}_1 = \{1, 2, \dots, n\}$ and $\mathcal{S}_2 = \{n+1, n+2, \dots, kn\}$. The alignment matrix A for graphs G_1 and G_2 can be characterized using equation (3.2) as follows:

$$A(t_1, t_2) = \begin{cases} (\alpha + 1)G_1(i, j)G_2(i', j') - G_1(i, j) - G_2(i', j') + 1 + \epsilon, \\ \text{if } t_1 \sim (i, i'), t_2 \sim (j, j'), t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2. \\ (\alpha + 1)G_1(i, j)G_2(r', s') - G_1(i, j) - G_2(r', s') + 1 + \epsilon \\ \text{if } t_1 \sim (i, r'), t_2 \sim (j, s'), t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2. \\ 1 + \epsilon, \\ \text{if } t_1 = t_2, \end{cases} \quad (9.4)$$

where notation $t_1 \sim (i, r')$ means that, row (and column) index t_1 of the alignment matrix A corresponds to the mapping edge (i, r') . Since G_1 and G_2 are isomorphic with permutation matrix $P = I$, we have $G_1(i, j) = G_2(i', j')$. Therefore, equation (9.4) can be written as,

$$A(t_1, t_2) = \begin{cases} (\alpha + 1)G_1(i, j)^2 - 2G_1(i, j) + 1 + \epsilon \\ \text{if } t_1 \sim (i, i'), t_2 \sim (j, j'), t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2. \\ (\alpha + 1)G_1(i, j)G_1(r, s) - G_1(i, j) - G_1(r, s) + 1 + \epsilon \\ \text{if } t_1 \sim (i, r'), t_2 \sim (j, s'), t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2. \\ 1 + \epsilon, \\ \text{if } t_1 = t_2. \end{cases} \quad (9.5)$$

Let \bar{A} be the expected alignment matrix, where $\bar{A}(t_1, t_2) = \mathbb{E}[A(t_1, t_2)]$, the expected value of $A(t_1, t_2)$ over different realizations of G_1 and G_2 .

Lemma 3 Let \mathbf{v} be the eigenvector of the expected alignment matrix \bar{A} corresponding to the largest eigenvalue. Suppose

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix},$$

where \mathbf{v}_1 and \mathbf{v}_2 are vectors of length n and $(k-1)n$, respectively. Then,

$$\begin{aligned} v_{1,1} = v_{1,2} = \dots = v_{1,n} &\triangleq v_1^*, \\ v_{2,1} = v_{2,2} = \dots = v_{2,(k-1)n} &\triangleq v_2^*, \end{aligned}$$

Moreover, if $n \rightarrow \infty$, then,

$$\frac{v_1^*}{v_2^*} > 1 + \Delta, \quad (9.6)$$

where $0 < \Delta k < \frac{(\alpha-1)p+1+\epsilon}{(\alpha+1)p^2-2p+1+\epsilon} - 1$.

Proof Since $G_1(i, j)$ is a Bernoulli random variable which is one with probability p , equation (9.5) leads to:

$$\bar{A}(t_1, t_2) = \begin{cases} (\alpha-1)p+1+\epsilon, & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2, \\ (\alpha+1)p^2-2p+1+\epsilon, & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2, \\ 1+\epsilon, & \text{if } t_1 = t_2. \end{cases} \quad (9.7)$$

Define $a \triangleq (\alpha-1)p+1+\epsilon$ and $b \triangleq (\alpha+1)p^2-2p+1+\epsilon$. Since $b\mathbf{v}$ is an eigenvector of \bar{A} , we have,

$$\bar{A}\mathbf{v} = \lambda\mathbf{v}, \quad (9.8)$$

where λ is the corresponding eigenvalue of $b\mathbf{v}$. Therefore,

$$\bar{A}\mathbf{v} = \begin{bmatrix} a \sum_i v_{1,i} + b \sum_j v_{2,j} + (1+\epsilon-a)v_{1,1} \\ \vdots \\ a \sum_i v_{1,i} + b \sum_j v_{2,j} + (1+\epsilon-a)v_{1,n} \\ b \sum_i v_{1,j} + b \sum_j v_{2,j} + (1+\epsilon-a)v_{2,1} \\ \vdots \\ b \sum_i v_{1,i} + b \sum_j v_{2,j} + (1+\epsilon-a)v_{2,(k-1)n} \end{bmatrix} = \lambda \begin{bmatrix} v_{1,1} \\ \vdots \\ v_{1,n} \\ v_{2,1} \\ \vdots \\ v_{2,(k-1)n} \end{bmatrix}. \quad (9.9)$$

Therefore,

$$\begin{aligned} a \sum_i v_{1,i} + b \sum_j v_{2,j} &= v_{1,r}(\lambda + a - 1 - \epsilon), & \forall 1 \leq r \leq n, \\ b \sum_i v_{1,i} + b \sum_j v_{2,j} &= v_{2,s}(\lambda + b - 1 - \epsilon), & \forall 1 \leq s \leq (k-1)n. \end{aligned} \quad (9.10)$$

We choose ϵ so that $\lambda + a - 1 - \epsilon \neq 0$ and $\lambda + b - 1 - \epsilon \neq 0$. We will show later in this section that any sufficiently small value of ϵ satisfies these inequalities. Therefore, equation (9.10) leads to,

$$\begin{aligned} v_{1,1} = v_{1,2} = \dots = v_{1,n} &= v_1^*, \\ v_{2,1} = v_{2,2} = \dots = v_{2,(k-1)n} &= v_2^*. \end{aligned} \quad (9.11)$$

Using equations (9.10) and (9.11), we have,

$$\begin{cases} anv_1^* + b(k-1)nv_2^* = v_1^*(\lambda + a - 1 - \epsilon) \\ bnv_1^* + b(k-1)nv_2^* = v_2^*(\lambda + b - 1 - \epsilon). \end{cases} \quad (9.12)$$

We choose ϵ so that $\lambda + b(1 - (k-1)n) - 1 - \epsilon \neq 0$. We will show later in this section that any sufficiently small value of ϵ satisfies this inequality. Further, according to the Perron-Frobenius Theorem 1, $v_{1,i} > 0$ and $v_{2,j} > 0$, for all i and j . Under these conditions, solving equation (9.12) leads to:

$$(\lambda - \lambda_a)(\lambda - \lambda_b) = b^2(k-1)n^2, \quad (9.13)$$

where,

$$\begin{cases} \lambda_a = (n-1)a + 1 + \epsilon, \\ \lambda_b = ((k-1)n-1)b + 1 + \epsilon. \end{cases} \quad (9.14)$$

Equation (9.13) has two solutions for λ . However, since λ is the largest eigenvalue of the expected alignment matrix \bar{A} , we choose the largest solution. Note that, since $b^2(k-1)n^2 > 0$, we have $\lambda > \max(\lambda_a, \lambda_b)$. This guarantees conditions that we put on ϵ in the early steps of the proof.

By solving equations (9.13) and (9.14), we have,

$$\lambda = \frac{\lambda_a + \lambda_b + \sqrt{(\lambda_a - \lambda_b)^2 + 4(k-1)b^2n^2}}{2}.$$

First, we show $v_1^* > v_2^*$. As $n \rightarrow \infty$, equation (9.12) implies,

$$\frac{v_1^*}{v_2^*} = \frac{\lambda}{bn} - k + 1, \quad (9.15)$$

where λ is the largest root of equation (9.13). For sufficiently large n ,

$$\frac{v_1^*}{v_2^*} = \frac{1}{2} \left[\left(\frac{a}{b} - k + 1 \right) + \sqrt{\left(\frac{a}{b} - k + 1 \right)^2 + 4k - 4} \right]. \quad (9.16)$$

If $p \neq 0, 1$, we always have $a > b$. Therefore, there exists $\Delta > 0$ such that $\frac{a}{b} > 1 + \Delta k$. Thus, we have,

$$\frac{a}{b} > 1 + \Delta k > 1 + \Delta \left(1 + \frac{k-1}{1+\Delta} \right) = 1 + \Delta + \frac{\Delta}{\Delta+1} (k-1). \quad (9.17)$$

Using inequality (9.17) in (9.16), we have,

$$\begin{aligned} \frac{v_1^*}{v_2^*} &> \frac{1}{2} \left[\frac{(1+\Delta)^2 - k + 1}{1+\Delta} \right. \\ &\quad \left. + \frac{\sqrt{((1+\Delta)^2 - k + 1)^2 + 4(k-1)(1+\Delta)^2}}{1+\Delta} \right] = 1 + \Delta. \end{aligned} \quad (9.18)$$

This completes the proof of Lemma 3. ■

9.3 Proof of Theorem 2

Without loss of generality and similarly to the proof of Theorem 1, let $P = I$. Let A be the alignment graph of G_1 and G_2 defined according to equation (3.2). Similarly to the proof of Theorem 1, reorder row (and column) indices of matrix A so that the first n indices correspond to the true mappings $\{(i, i') : i \in \mathcal{V}_1, i' \in \mathcal{V}_2\}$. Define the expected alignment graph \bar{A} as $\bar{A}(t_1, t_2) = \mathbb{E}[A(t_1, t_2)]$, where t_1 and t_2 are two possible mappings across graphs. Recall notations $\mathcal{S}_1 = \{1, 2, \dots, n\}$ and $\mathcal{S}_2 = \{n+1, n+2, \dots, kn\}$.

First, we consider the noise model I (3.9). Define,

$$\begin{aligned} a' &\triangleq p(1-p_e)(\alpha + \epsilon) + (1-p)(1-p_e)(1 + \epsilon) + (pp_e + (1-p)p_e)\epsilon \\ b' &\triangleq (p^2(1-p_e) + pp_e(1-p))(\alpha + \epsilon) \\ &\quad + ((1-p)^2(1-p_e) + pp_e(1-p))(1 + \epsilon) \\ &\quad + (2p(1-p)(1-p_e) + 2p^2p_e)\epsilon. \end{aligned} \quad (9.19)$$

Since $G_1(i, j)$ and $Q(i, j)$ are Bernoulli random variables with parameters p and p_e , respectively, the expected alignment graph can be simplified as follows:

$$\bar{A}(t_1, t_2) = \begin{cases} a', & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2, \\ b', & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2, \\ 1 + \epsilon, & \text{if } t_1 = t_2. \end{cases} \quad (9.20)$$

We have,

$$a' - b' = (\alpha + 1)(2p_e - 1)p(p - 1) + p_e(1 - 2p)\epsilon. \quad (9.21)$$

Thus, if $p \neq 0, 1$ and $p_e < 1/2$, for small enough ϵ , $a' > b' > 0$. Therefore, there exists a positive Δ such that $\frac{a'}{b'} = 1 + \Delta$. The rest of the proof is similar to the one of Theorem 1.

The proof for the noise model II of (3.10) is similar. To simplify notation and illustrate the main idea, here we assume ϵ is sufficiently small with negligible effects. Define,

$$\begin{aligned} a'' &\triangleq p(1-p_e)(\alpha) + (1-p)(1-p_{e_2}) = 1 - p(1 + \alpha(p_e - 1) + p_e) \\ b'' &\triangleq p^2(1-p_e)\alpha + (1-p)^2(1-p_{e_2}) + 2p(1-p)p_{e_2}(1 + \alpha) \\ &\quad = 1 - p(2 + p_e) + p^2(1 + \alpha + 2p_e). \end{aligned} \quad (9.22)$$

The expected alignment graph in this case is:

$$\bar{A}(t_1, t_2) = \begin{cases} a'', & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2, \\ b'', & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2, \\ 1 + \epsilon, & \text{if } t_1 = t_2. \end{cases} \quad (9.23)$$

Moreover, we have,

$$a'' - b'' = p((1-p-p_e)(1 + \alpha) + p_e(1 - 2p)). \quad (9.24)$$

If $p < 1/2$ and $p_e < 1/2$, then $a'' - b'' > 0$. The rest of the proof is similar to the previous case.

9.4 Proof Of Theorem 4

By writing Taylor's expansion of $Tr(G_1 X G_2 X^T)$ around the point X_0 , we have,

$$\begin{aligned} Tr(G_1 X G_2 X^T) = & Tr(G_1 X_0 G_2 X_0^T) + 2Tr(G_1 X_0 G_2 (X - X_0)^T) \\ & + Tr(G_1 (X - X_0) G_2 (X - X_0)^T). \end{aligned} \quad (9.25)$$

Let $\Delta \triangleq X - X_0$. Thus, we have,

$$f(X) = \tilde{f}(X) + Tr(G_1 \Delta G_2 \Delta^T). \quad (9.26)$$

Let $\sigma_i(G)$ be the i -th largest singular value of matrix G .

Theorem 5 (Von Neumann's trace inequality) *Suppose A and B are two $n \times n$ complex matrices. We have,*

$$|Tr(AB)| \leq \sum_{i=1}^n \sigma_i(A) \sigma_i(B). \quad (9.27)$$

Proof See Theorem 1 of [68]. ■

Using Theorem 5, we have,

$$|Tr(G_1 \Delta G_2 \Delta^T)| \leq \sum_{i=1}^n \sigma_i(G_1 \Delta) \sigma_i(G_2 \Delta^T). \quad (9.28)$$

Moreover, we have,

$$\begin{aligned} \sigma_i(G_1 \Delta) & \leq \min\{\sigma_k(G_1) \sigma_{i+1-k}(\Delta) : 1 \leq k \leq i\} \\ & \leq \sigma_i(G_1) \sigma_1(\Delta). \end{aligned} \quad (9.29)$$

Moreover,

$$\begin{aligned} \sigma_i(G_1 \Delta) & \geq \max\{\sigma_k(G_1) \sigma_{n+i-k}(\Delta) : i \leq k \leq n\}, \\ & \geq \sigma_i(G_1) \sigma_n(\Delta). \end{aligned} \quad (9.30)$$

Moreover, since $\|\Delta\|_2 \leq \epsilon$, we have $|\sigma_i(\Delta)| \leq \epsilon$, for $1 \leq i \leq n$. Using (9.29) and (9.30), we have,

$$|\sigma_i(G_1 \Delta)| \leq \epsilon \sigma_i(G_1). \quad (9.31)$$

Similarly, we have,

$$|\sigma_i(G_2 \Delta^T)| \leq \epsilon \sigma_i(G_2). \quad (9.32)$$

Thus, using (9.28) and (9.32), we have,

$$|Tr(G_1 \Delta G_2 \Delta^T)| \leq \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2). \quad (9.33)$$

Using (9.26) and (9.33), we have,

$$|f(X) - \tilde{f}(X)| \leq \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2). \quad (9.34)$$

Moreover, since X^* and X_{lin}^* are optimal solutions of optimizations (4.1) and (4.7), respectively, we have,

$$\begin{aligned} f(X^*) &\leq \tilde{f}(X^*) + \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2) \\ &\leq \tilde{f}(X_{lin}^*) + \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2) \\ f(X^*) &\geq f(X_{lin}^*) \geq \tilde{f}(X_{lin}^*) - \epsilon^2 \sum_{i=1}^n \sigma_i(G_1) \sigma_i(G_2). \end{aligned} \quad (9.35)$$

This completes the proof.

Appendix

A Inference of Regulatory Networks

In this section we leverage genome-wide functional genomics datasets from ENCODE and modENCODE consortia to infer regulatory networks across human, fly, and worm. In Section 6 in the main text, we compare the structure of these inferred networks using network alignment techniques.

The temporal and spatial expression of genes is coordinated by a hierarchy of transcription factors (TFs), whose interactions with each other and with their target genes form directed regulatory networks [69]. In addition to individual interactions, the structure of a regulatory network captures a broad systems-level view of regulatory and functional processes, since genes cluster into modules that perform similar functions [70–72]. Accurate inference of these regulatory networks is important both in the recovery and functional characterization of gene modules, and for comparative genomics of regulatory networks across multiple species [73, 74]. This is especially important because animal genomes, as fly, worm, and mouse are routinely used as models for human disease [75, 76].

Here, we infer regulatory networks of human, and model organisms *D. melanogaster* fly, and *C. elegans* worm, three of the most distant and deeply studied metazoan species. To infer regulatory interactions among transcription factors and target genes in each species, we combine genome-wide transcription factor binding profiles, conserved sequence motif instances [77] and gene expression levels [78, 79] for multiple cell types that have been collected by the ENCODE and modENCODE consortia. The main challenge is to integrate these diverse evidence sources of gene regulation in order to infer robust and accurate regulatory interactions for each species.

Ideally, inference of regulatory networks would involve performing extensive all-against-all experiments of chromatin immune-precipitation (ChIP) assays for every known transcription factor in every cell type of an organism, in order to identify all potential targets of TFs, followed by functional assays to verify that a TF-gene interaction is functional [72, 80]. However, the combinatorial number of pairs of TFs and cell types makes this experiment prohibitively expensive, necessitating the use of methods to reduce dimensionality of this problem. Here, we first infer three types of feature-specific regulatory connections based on functional and physical evidences and then integrate them to infer regulatory interactions in each species (Figure 5-a). One feature-specific network is based on using sequence motifs to scan the genome for instances of known binding sites of each TF, and then match predicted binding instances to nearby genes (a motif network). A second approach is to map TFs to genes nearby their ChIP peaks using a window-based approach (a ChIP binding network). The third feature specific network uses gene expression profiles under different conditions in order to find groups of genes that are correlated in expression and therefore likely to function together (an expression-based network).

Previous work [72] has shown that, while ChIP networks are highly informative of true regulatory interactions, the number of experiments that can be carried out is typically very small, yielding a small number of high confidence interactions. Motif networks tend to be less informative than ChIP networks, but yield more coverage of the regulatory network, while co-expression based networks tend to include many false-positive edges and are the least informative [80, 81]. However, integration of these three networks [71, 82–84] into a combined network yield better performance than the individual networks in terms of recovering known regulatory interactions, by predicting interactions that tend to be supported by multiple lines of evidence. Here, we use an integration approach that combines interaction ranks across networks [84]. Inferred regulatory interactions show

significant overlap with known interactions in human and fly, indicating the accuracy and robustness of the used inference pipeline. In the following, we explain our network inference framework with more details.

A.1 Inference of feature specific regulatory networks

For each species, we form feature-specific regulatory networks using functional (gene expression profiles) and physical (motif sequences and ChIP peaks) evidences as follows:

Functional association networks. Expression-based networks represent interactions among TFs and target genes which are supported by correlation in gene expression levels across multiple samples [69, 85–87]. There are several methods to infer regulatory networks using gene expression profiles [84]. The input for these algorithms is a gene by condition matrix of expression values. The output of these methods are expression-based regulatory networks. We use the side information of TF lists to remove outgoing edges from target genes (in fact, TF lists are used as inputs to network inference algorithms to enhance their performance by limiting search space of the methods.).

To reduce bias and obtain a single expression-based network for each species, we combine results of two different expression-based network inference methods (Figure 5-a): one method is CLR [78] (context likelihood of relatedness) which constructs expression networks using mutual information among gene expression profiles along with a correction step to eliminate background correlations. The second method used is GENIE3 [79] (Gene Network Inference with Ensemble of Trees) which is a tree-based ensemble method that decomposes the network inference problem to several feature selection subproblems. In each subproblem, it identifies potential regulators by performing a regression analysis using random forest. GENIE3 has been recognized as the top-performing expression based inference method in the DREAM5 challenge [84].

Table 1 summarizes the number of genes and TFs in expression-based regulatory networks. These numbers refer to genes and TFs that are mapped to Entrez Gene IDs [88], the standard IDs that we use throughout our analysis. As it is illustrated in this table, expression based networks cover most of potential regulatory edges from TFs to targets. Despite a high coverage, however, the quality of inferred expression networks are lower than the one for physical networks [80]. This can be partially owing to indirect effects and transitive interactions in expression-based regulatory networks [84].

Physical association networks. We form two physical regulatory networks for each of the considered species using two types of physical evidences as our inference features: In the first approach, we use conserved occurrences of known sequence motifs [77], while in the second approach, we use experimentally defined TF binding occupancy profiles from ChIP assays of ENCODE and modENCODE [72, 80]. Table 2 shows the number of TFs associated to motifs as well as the number of TFs with genome-wide ChIP profiles in human, fly and worm. TSS coordinates are based on the genome annotations from ENCODE and modENCODE for human and worm, respectively, and the FlyBase genome annotations (FB5.48) for fly.

Each physical feature is assigned to a score: motif sequence features are assigned to conservation scores according to a phylogenetic framework [77], while sequence read density of TFs determines scores of ChIP peaks. Further, two physical features are called overlapping if their corresponding sequences have a minimum overlap of 25% in relation to their lengths (Jaccard Index > 0.25).

Our inference algorithm is based on occurrence of these features (motif sequences or ChIP peaks) within a fixed window size around the transcription start site (TSS) of target genes (Figure 5-b). We use a fixed window of 5kb around the transcription start site (TSS) in human and 1kb

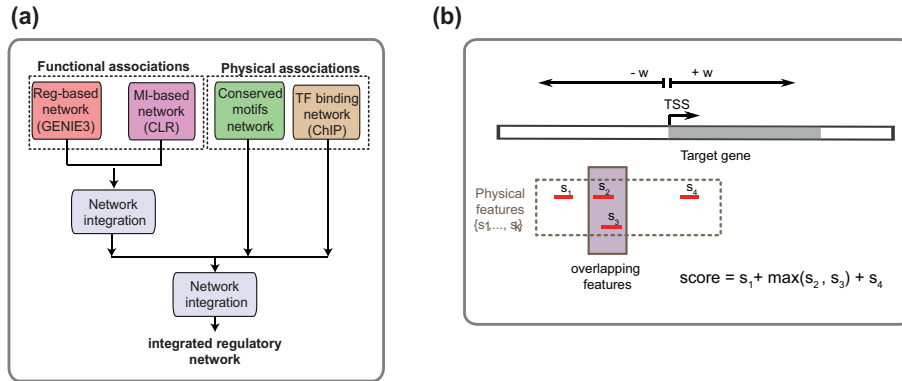


Figure 5: (a) The proposed framework to infer integrative regulatory networks. (b) The proposed framework to infer physical feature-specific regulatory networks.

in fly and worm. Then, we apply a max-sum algorithm to assign weights to TF-target interactions in each case: we take the maximum score of overlapping features and sum the scores of non-overlapping ones. In ChIP networks, because read densities are not comparable across different TFs, we normalize TF-target weights for each TF by computing z-scores.

A.2 Inference of integrated regulatory networks

Feature specific networks have certain biases and shortcomings. While Physical networks (motif and ChIP networks) show high quality considering overlap of their interactions with known interactions [80], their coverage of the entire network is pretty low mostly owing to the cost of the experiments. On the other hand, while expression based networks have a larger coverage of regulatory networks compared to physical ones, they include many false-positive edges partially owing to indirect information flows [81]. To overcome these limitations, we therefore integrate these feature-specific regulatory interactions into a single integrated network [71, 82–84] (Figure 5-a).

Suppose there are K input feature-specific regulatory networks, each with n genes and m TFs (only TF nodes can have out-going edges in the network). Let $w_{i,j}^l$ and $w_{i,j}$ represent interaction weights between TF i and target gene j in the input network l and in the integrative network, respectively. We use a rank-based (borda) integration technique to infer integrated networks in considered species. In this approach, integrative weights are computed as follows:

$$w_{i,j} = 1/K \sum_{l=1}^K r_{i,j}^l, \quad (\text{A.1})$$

where $r_{i,j}^l$ represents the rank of interactions $i \rightarrow j$ in the input network l . An edge with the maximum weight is mapped to the rank nm . We also assume non-existent interactions are mapped to rank 0 (if $w_{i,j}^l = 0$, then $r_{i,j}^l = 0$) [84]. Moreover, ties are broken randomly among edges with same weights.

We find that top-ranking integrative interactions in human and fly networks are primarily supported by ChIP and motif evidences, while worm interactions are primarily supported by co-expression edges, consistent with the lower coverage of worm ChIP and motif interactions (Figure 6).

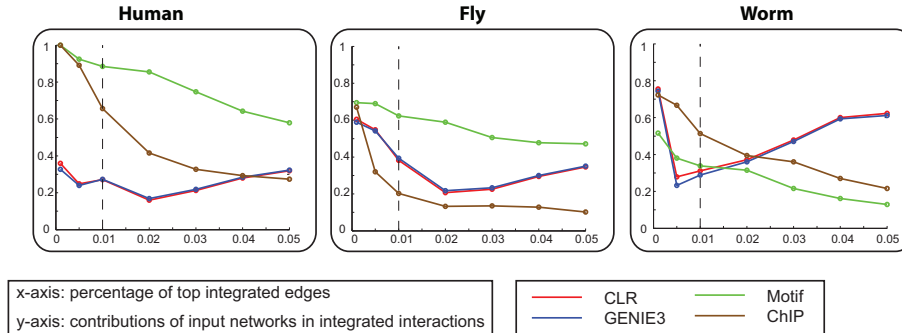


Figure 6: Contributions of input feature-specific networks in integrated interactions.

	Human	Fly	Worm
Genes	19,088	12,897	19,277
TFs	2,757	675	905

Table 1: Number of genes and TFs covered by gene expression data.

	Human	Fly	Worm
Motif network	485	221	30
ChIP network	165	51	88

Table 2: Number of TFs covered by evolutionary conserved motifs and TF binding datasets.

To validate inferred integrated networks, we use known interactions in TRANSFAC [89], RED-fly [90] and EdgeDB [91] as human, fly and worm benchmarks, respectively. We assess the quality of various networks by using (a) the area under the receiver operating characteristic curve (AUROC); and (b) the area under the precision recall curve (AUPR), for each benchmark network (Figures 7). Let $TP(k)$ and $FP(k)$ represent the number of true positives and false positives in top k predictions, respectively. Suppose the total number of positives and negatives in the gold standard are represented by P and N , respectively. Then, an ROC curve plots true positive rate vs. false positive rate ($TP(k)/P$ vs. $FP(k)/N$), while a PR curve plots precision ($TP(k)/k$) vs. recall ($TP(k)/P$). A high AUPR value indicates that, top predictions significantly overlap with known interactions, while a high AUROC value indicates the advantage of inferred predictions in discriminating true and false positives compared to random predictions (AUROC of a random predictor is 0.5).

Figure 7 illustrates AUROC and AUPR scores for feature-specific and integrative networks, in different cut-offs, and in all three considered species. Considering the top 5% of interactions in each weighted network as predicted edges, according to AUROC metric, integrative networks outperform feature-specific networks in all three species. In fact, AUROC values of integrative networks are 0.58 in human, 0.62 in fly, and 0.52 in worm, respectively. AUPR values of integrative networks are 0.019 in human, 0.047 in fly, and 0.037 in worm, respectively. Notably, all methods have low scores over the EdgeDB (worm) benchmark, which can be partially owing to sparse physical networks and/or systematic bias of EdgeDB interactions.

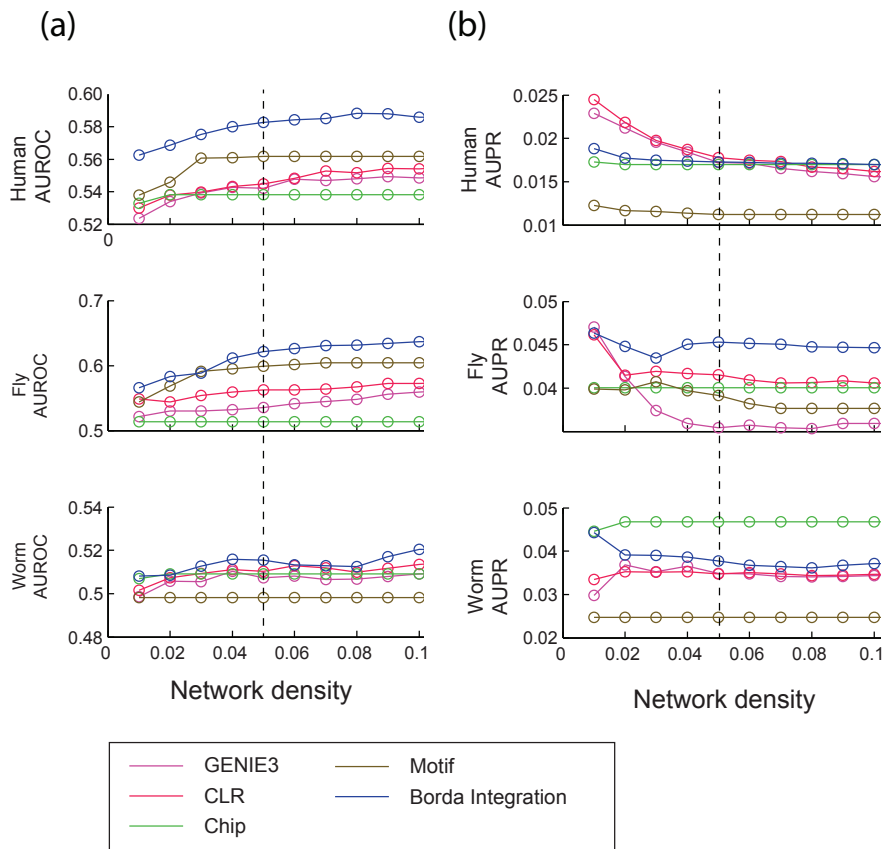


Figure 7: AUROC and AUPR scores of feature-specific and integrated regulatory networks in human, fly and worm species.

As the cut-off (network density) increases, AUROC values of integrative networks tend to increase while their AUPR scores are decreasing in general. This is because of the fact that, the rate of true positives is lower among medium ranked interactions compared to top ones. Considering both AUROC and AUPR curves for all species, we binarize networks using their top 5% interactions which leads to balanced values of AUROC and AUPR in all inferred networks. This results in $2.6M$ interactions in human, $469k$ in fly and $876k$ in worm. In integrative networks, the median number of targets for each TF is 253 in human, 290 in fly and 640 in worm, with a median of 132 regulators per gene in human, 29 in fly, and 43 in worm.

Unlike EigenAlign, other considered network alignment methods do not take into account the directionality of edges in their network alignment setup. Thus, to have fair performance assessments of considered network alignment methods, we create un-directed co-regulatory networks using inferred regulatory networks by connecting genes when their parent TFs have an overlap larger than 25%. This results in undirected binary networks in human, fly, and worm, with 19,221, 13,642, and 19,296 nodes, and 13.9%, 3.5%, and 4.2% edge densities, respectively.

References

- [1] R. Sharan and T. Ideker, “Modeling cellular machinery through biological network comparison,” *Nature biotechnology*, vol. 24, no. 4, pp. 427–433, 2006.
- [2] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*. Macmillan London, 1976, vol. 6.
- [3] R. Singh, J. Xu, and B. Berger, “Global alignment of multiple protein interaction networks with application to functional orthology detection,” *Proceedings of the National Academy of Sciences*, vol. 105, no. 35, pp. 12 763–12 768, 2008.
- [4] C.-S. Liao, K. Lu, M. Baym, R. Singh, and B. Berger, “Isorankn: spectral methods for global alignment of multiple protein networks,” *Bioinformatics*, vol. 25, no. 12, pp. i253–i258, 2009.
- [5] J. Flannick, A. Novak, B. S. Srinivasan, H. H. McAdams, and S. Batzoglou, “Graemlin: general and robust alignment of multiple large interaction networks,” *Genome research*, vol. 16, no. 9, pp. 1169–1181, 2006.
- [6] M. Zaslavskiy, F. Bach, and J.-P. Vert, “Global alignment of protein–protein interaction networks by graph matching methods,” *Bioinformatics*, vol. 25, no. 12, pp. i259–i267, 2009.
- [7] B. P. Kelley, B. Yuan, F. Lewitter, R. Sharan, B. R. Stockwell, and T. Ideker, “Pathblast: a tool for alignment of protein interaction networks,” *Nucleic acids research*, vol. 32, no. suppl 2, pp. W83–W88, 2004.
- [8] M. Kalaei, M. Smoot, T. Ideker, and R. Sharan, “Networkblast: comparative analysis of protein networks,” *Bioinformatics*, vol. 24, no. 4, pp. 594–596, 2008.
- [9] 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*, vol. 18, no. 03, pp. 265–298, 2004.
- [10] C. Schellewald and C. Schnörr, “Probabilistic subgraph matching based on convex relaxation,” in *Energy minimization methods in computer vision and pattern recognition*. Springer, 2005, pp. 171–186.
- [11] S. Lacoste-Julien, B. Taskar, D. Klein, and M. I. Jordan, “Word alignment via quadratic assignment,” in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 112–119.
- [12] S. Melnik, H. Garcia-Molina, and E. Rahm, “Similarity flooding: A versatile graph matching algorithm and its application to schema matching,” in *Data Engineering, 2002. Proceedings. 18th International Conference on*. IEEE, 2002, pp. 117–128.
- [13] A. Narayanan and V. Shmatikov, “De-anonymizing social networks,” in *Security and Privacy, 2009 30th IEEE Symposium on*. IEEE, 2009, pp. 173–187.
- [14] R. E. Burkard, *Quadratic assignment problems*. Springer, 2013.

- [15] K. Makarychev, R. Manokaran, and M. Sviridenko, “Maximum quadratic assignment problem: Reduction from maximum label cover and lp-based approximation algorithm,” in *Automata, Languages and Programming*. Springer, 2010, pp. 594–604.
- [16] M. Bazaraa and O. Kirca, “A branch-and-bound-based heuristic for solving the quadratic assignment problem,” *Naval research logistics quarterly*, vol. 30, no. 2, pp. 287–304, 1983.
- [17] M. S. Bazaraa and H. D. Sherali, “On the use of exact and heuristic cutting plane methods for the quadratic assignment problem,” *Journal of the Operational Research Society*, pp. 991–1003, 1982.
- [18] E. L. Lawler, “The quadratic assignment problem,” *Management science*, vol. 9, no. 4, pp. 586–599, 1963.
- [19] L. Kaufman and F. Broeckx, “An algorithm for the quadratic assignment problem using bender’s decomposition,” *European Journal of Operational Research*, vol. 2, no. 3, pp. 207–211, 1978.
- [20] A. Frieze and J. Yadegar, “On the quadratic assignment problem,” *Discrete applied mathematics*, vol. 5, no. 1, pp. 89–98, 1983.
- [21] W. P. Adams and T. A. Johnson, “Improved linear programming-based lower bounds for the quadratic assignment problem,” *DIMACS series in discrete mathematics and theoretical computer science*, vol. 16, pp. 43–75, 1994.
- [22] G. Finke, R. E. Burkard, and F. Rendl, “Quadratic assignment problems,” *North-Holland Mathematics Studies*, vol. 132, pp. 61–82, 1987.
- [23] S. Hadley, F. Rendl, and H. Wolkowicz, “A new lower bound via projection for the quadratic assignment problem,” *Mathematics of Operations Research*, vol. 17, no. 3, pp. 727–739, 1992.
- [24] K. Anstreicher and H. Wolkowicz, “On lagrangian relaxation of quadratic matrix constraints,” *SIAM Journal on Matrix Analysis and Applications*, vol. 22, no. 1, pp. 41–55, 2000.
- [25] K. M. Anstreicher and N. W. Brixius, “Solving quadratic assignment problems using convex quadratic programming relaxations,” *Optimization Methods and Software*, vol. 16, no. 1-4, pp. 49–68, 2001.
- [26] Q. Zhao, S. E. Karisch, F. Rendl, and H. Wolkowicz, “Semidefinite programming relaxations for the quadratic assignment problem,” *Journal of Combinatorial Optimization*, vol. 2, no. 1, pp. 71–109, 1998.
- [27] J. Peng, H. Mittelmann, and X. Li, “A new relaxation framework for quadratic assignment problems based on matrix splitting,” *Mathematical Programming Computation*, vol. 2, no. 1, pp. 59–77, 2010.
- [28] J. T. Vogelstein, J. M. Conroy, V. Lyzinski, L. J. Podrazik, S. G. Kratzer, E. T. Harley, D. E. Fishkind, R. J. Vogelstein, and C. E. Priebe, “Fast approximate quadratic programming for graph matching,” *PLOS one*, vol. 10, no. 4, 2015.

- [29] G. W. Klau, “A new graph-based method for pairwise global network alignment,” *BMC bioinformatics*, vol. 10, no. Suppl 1, p. S59, 2009.
- [30] M. El-Kebir, J. Heringa, and G. W. Klau, “Lagrangian relaxation applied to sparse global network alignment,” in *IAPR International Conference on Pattern Recognition in Bioinformatics*, 2011, pp. 225–236.
- [31] —, “Natalie 2.0: Sparse global network alignment as a special case of quadratic assignment,” *Algorithms*, vol. 8, no. 4, pp. 1035–1051, 2015.
- [32] M. Kolář, J. Meier, V. Mustonen, M. Lässig, and J. Berg, “Graphalignment: Bayesian pairwise alignment of biological networks,” *BMC systems biology*, vol. 6, no. 1, p. 144, 2012.
- [33] M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang, “Message-passing algorithms for sparse network alignment,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 7, no. 1, p. 3, 2013.
- [34] M. Leordeanu and M. Hebert, “A spectral technique for correspondence problems using pairwise constraints,” in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1482–1489.
- [35] M. Carcassoni and E. R. Hancock, “Alignment using spectral clusters.” in *BMVC*, 2002, pp. 1–10.
- [36] E. Kazemi, H. S Hamed, and M. Grossglauser, “Growing a graph matching from a handful of seeds,” in *Proceedings of the Vldb Endowment International Conference on Very Large Data Bases*, vol. 8, no. EPFL-ARTICLE-207759, 2015.
- [37] C. Clark and J. Kalita, “A multiobjective memetic algorithm for ppi network alignment,” *Bioinformatics*, p. btv063, 2015.
- [38] N. Malod-Dognin and N. Pržulj, “L-graal: Lagrangian graphlet-based network aligner,” *Bioinformatics*, p. btv130, 2015.
- [39] E. M. Loiola, N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, “A survey for the quadratic assignment problem,” *European Journal of Operational Research*, vol. 176, no. 2, pp. 657–690, 2007.
- [40] F. Emmert-Streib, M. Dehmer, and Y. Shi, “Fifty years of graph matching, network alignment and network comparison,” *Information Sciences*, vol. 346, pp. 180–197, 2016.
- [41] M. E. Newman, “Modularity and community structure in networks,” *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.
- [42] D. L. Sussman, M. Tang, D. E. Fishkind, and C. E. Priebe, “A consistent adjacency spectral embedding for stochastic blockmodel graphs,” *Journal of the American Statistical Association*, vol. 107, no. 499, pp. 1119–1128, 2012.
- [43] T. Qin and K. Rohe, “Regularized spectral clustering under the degree-corrected stochastic blockmodel,” in *Advances in Neural Information Processing Systems*, 2013, pp. 3120–3128.

- [44] A. Athreya, V. Lyzinski, D. J. Marchette, C. E. Priebe, D. L. Sussman, and M. Tang, “A central limit theorem for scaled eigenvectors of random dot product graphs,” *arXiv preprint arXiv:1305.7388*, 2013.
- [45] A. Saade, F. Krzakala, and L. Zdeborová, “Spectral clustering of graphs with the bethe hessian,” in *Advances in Neural Information Processing Systems*, 2014, pp. 406–414.
- [46] C. Schellewald, S. Roth, and C. Schnörr, “Evaluation of a convex relaxation to a quadratic assignment matching approach for relational object views,” *Image and Vision Computing*, vol. 25, no. 8, pp. 1301–1314, 2007.
- [47] R. Patro and C. Kingsford, “Global network alignment using multiscale spectral signatures,” *Bioinformatics*, vol. 28, no. 23, pp. 3105–3114, 2012.
- [48] P. Erdős and A. Rényi, “On the strength of connectedness of a random graph,” *Acta Mathematica Hungarica*, vol. 12, no. 1, pp. 261–267, 1961.
- [49] T. Czajka and G. Pandurangan, “Improved random graph isomorphism,” *Journal of Discrete Algorithms*, vol. 6, no. 1, pp. 85–92, 2008.
- [50] L. Babai, “Graph isomorphism in quasipolynomial time [extended abstract],” in *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2016, pp. 684–697.
- [51] C. Clark and J. Kalita, “A comparison of algorithms for the pairwise alignment of biological networks,” *Bioinformatics*, vol. 30, no. 16, pp. 2351–2359, 2014.
- [52] W. Ali and C. M. Deane, “Functionally guided alignment of protein interaction networks for module detection,” *Bioinformatics*, vol. 25, no. 23, pp. 3166–3173, 2009.
- [53] S. Mohammadi, D. Gleich, T. Kolda, and A. Grama, “Triangular alignment (tame): A tensor-based approach for higher-order network alignment,” *arXiv preprint arXiv:1510.06482*, 2015.
- [54] P. Schweitzer, “Problems of unknown complexity: graph isomorphism and ramsey theoretic numbers,” Ph.D. dissertation, Saarbrücken, Univ., Diss., 2009, 2009.
- [55] L. Babai and L. Kucera, “Canonical labelling of graphs in linear average time,” in *Foundations of Computer Science, 1979., 20th Annual Symposium on*. IEEE, 1979, pp. 39–46.
- [56] L. N. Trefethen and D. Bau III, *Numerical linear algebra*. Siam, 1997, vol. 50.
- [57] S. U. Pillai, T. Suel, and S. Cha, “The perron-frobenius theorem: some of its applications,” *IEEE Signal Processing Magazine*, vol. 22, no. 2, pp. 62–75, 2005.
- [58] D. B. West *et al.*, *Introduction to graph theory*. Prentice hall Upper Saddle River, 2001, vol. 2.
- [59] J. Kuczynski and H. Wozniakowski, “Estimating the largest eigenvalue by the power and lanczos algorithms with a random start,” *SIAM journal on matrix analysis and applications*, vol. 13, no. 4, pp. 1094–1122, 1992.

- [60] R. Preis, “Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs,” in *Annual Symposium on Theoretical Aspects of Computer Science*, 1999, pp. 259–269.
- [61] C. Fraikin, Y. Nesterov, and P. Van Dooren, “A gradient-type algorithm optimizing the coupling between matrices,” *Linear Algebra and its Applications*, vol. 429, no. 5, pp. 1229–1242, 2008.
- [62] —, “Optimizing the coupling between two isometric projections of matrices,” *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 1, pp. 324–345, 2008.
- [63] D. Knossow, A. Sharma, D. Mateus, and R. Horaud, “Inexact matching of large and sparse graphs using laplacian eigenvectors,” in *International workshop on graph-based representations in pattern recognition*, 2009, pp. 144–153.
- [64] S. Umeyama, “An eigendecomposition approach to weighted graph matching problems,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, no. 5, pp. 695–703, 1988.
- [65] R. Kohavi *et al.*, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *IJCAI*, vol. 14, no. 2, 1995, pp. 1137–1145.
- [66] W. Aiello, F. Chung, and L. Lu, “A random graph model for power law graphs,” *Experimental Mathematics*, vol. 10, no. 1, pp. 53–66, 2001.
- [67] A. P. Boyle, C. L. Araya, C. Brdlik, P. Cayting, C. Cheng, Y. Cheng, K. Gardner, L. W. Hillier, J. Janette, L. Jiang *et al.*, “Comparative analysis of regulatory information and circuits across distant species,” *Nature*, vol. 512, no. 7515, pp. 453–456, 2014.
- [68] L. Mirsky, “A trace inequality of john von neumann,” *Monatshefte für mathematik*, vol. 79, no. 4, pp. 303–306, 1975.
- [69] R. De Smet and K. Marchal, “Advantages and limitations of current network inference methods,” *Nature Reviews Microbiology*, vol. 8, no. 10, pp. 717–729, 2010.
- [70] E. Segal, M. Shapira, A. Regev, D. Pe’er, D. Botstein, D. Koller, and N. Friedman, “Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data,” *Nature genetics*, vol. 34, no. 2, pp. 166–176, 2003.
- [71] Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young *et al.*, “Computational discovery of gene modules and regulatory networks,” *Nature biotechnology*, vol. 21, no. 11, pp. 1337–1342, 2003.
- [72] D. Marbach, S. Roy, F. Ay, P. E. Meyer, R. Candeias, T. Kahveci, C. A. Bristow, and M. Kellis, “Predictive regulatory models in drosophila melanogaster by integrative inference of transcriptional networks,” *Genome research*, vol. 22, no. 7, pp. 1334–1349, 2012.
- [73] R. Sharan and T. Ideker, “Modeling cellular machinery through biological network comparison,” *Nature biotechnology*, vol. 24, no. 4, pp. 427–433, 2006.

- [74] S. A. McCarroll, C. T. Murphy, S. Zou, S. D. Pletcher, C.-S. Chin, Y. N. Jan, C. Kenyon, C. I. Bargmann, and H. Li, “Comparing genomic expression patterns across species identifies shared transcriptional profile in aging,” *Nature genetics*, vol. 36, no. 2, pp. 197–204, 2004.
- [75] J. O. Woods, U. M. Singh-Blom, J. M. Laurent, K. L. McGary, and E. M. Marcotte, “Prediction of gene–phenotype associations in humans, mice, and plants using phenologs,” *BMC bioinformatics*, vol. 14, no. 1, p. 203, 2013.
- [76] V. R. Chintapalli, J. Wang, and J. A. Dow, “Using flyatlas to identify better drosophila melanogaster models of human disease,” *Nature genetics*, vol. 39, no. 6, pp. 715–720, 2007.
- [77] P. Kheradpour, A. Stark, S. Roy, and M. Kellis, “Reliable prediction of regulator targets using 12 drosophila genomes,” *Genome research*, vol. 17, no. 12, pp. 1919–1931, 2007.
- [78] J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner, “Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles,” *PLoS biology*, vol. 5, no. 1, p. e8, 2007.
- [79] A. Irrthum, L. Wehenkel, P. Geurts *et al.*, “Inferring regulatory networks from expression data using tree-based methods,” *PloS one*, vol. 5, no. 9, p. e12776, 2010.
- [80] S. Roy, J. Ernst, P. V. Kharchenko, P. Kheradpour, N. Negre, M. L. Eaton, J. M. Landolin, C. A. Bristow, L. Ma, M. F. Lin *et al.*, “Identification of functional elements and regulatory circuits by drosophila modencode,” *Science*, vol. 330, no. 6012, pp. 1787–1797, 2010.
- [81] S. Feizi, D. Marbach, M. Médard, and M. Kellis, “Network deconvolution as a general method to distinguish direct dependencies in networks,” *Nature biotechnology*, 2013.
- [82] D. J. Reiss, N. S. Baliga, and R. Bonneau, “Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks,” *BMC bioinformatics*, vol. 7, no. 1, p. 280, 2006.
- [83] A. Greenfield, A. Madar, H. Ostrer, and R. Bonneau, “Dream4: Combining genetic and dynamic information to identify biological networks and dynamical models,” *PloS one*, vol. 5, no. 10, p. e13397, 2010.
- [84] D. Marbach, J. C. Costello, R. Küffner, N. M. Vega, R. J. Prill, D. M. Camacho, K. R. Allison, M. Kellis, J. J. Collins, G. Stolovitzky *et al.*, “Wisdom of crowds for robust gene network inference,” *Nature methods*, vol. 9, no. 8, pp. 796–804, 2012.
- [85] D. Marbach, R. J. Prill, T. Schaffter, C. Mattiussi, D. Floreano, and G. Stolovitzky, “Revealing strengths and weaknesses of methods for gene network inference,” *Proceedings of the National Academy of Sciences*, vol. 107, no. 14, pp. 6286–6291, 2010.
- [86] R. Bonneau, D. J. Reiss, P. Shannon, M. Facciotti, L. Hood, N. S. Baliga, and V. Thorsson, “The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo,” *Genome biology*, vol. 7, no. 5, p. R36, 2006.
- [87] N. Friedman, M. Linial, I. Nachman, and D. Pe’er, “Using bayesian networks to analyze expression data,” *Journal of computational biology*, vol. 7, no. 3-4, pp. 601–620, 2000.

- [88] D. Maglott, J. Ostell, K. D. Pruitt, and T. Tatusova, “Entrez gene: gene-centered information at ncbi,” *Nucleic acids research*, vol. 33, no. suppl 1, pp. D54–D58, 2005.
- [89] E. Wingender, X. Chen, R. Hehl, H. Karas, I. Liebich, V. Matys, T. Meinhardt, M. Prüß, I. Reuter, and F. Schacherer, “Transfac: an integrated system for gene expression regulation,” *Nucleic acids research*, vol. 28, no. 1, pp. 316–319, 2000.
- [90] S. M. Gallo, D. T. Gerrard, D. Miner, M. Simich, B. Des Soye, C. M. Bergman, and M. S. Halfon, “Redfly v3. 0: toward a comprehensive database of transcriptional regulatory elements in drosophila,” *Nucleic acids research*, vol. 39, no. suppl 1, pp. D118–D123, 2011.
- [91] M. I. Barrasa, P. Vaglio, F. Cavasino, L. Jacotot, and A. J. Walhout, “Edgedb: a transcription factor-dna interaction database for the analysis of *c. elegans* differential gene expression,” *BMC genomics*, vol. 8, no. 1, p. 21, 2007.