Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2015-005 February 18, 2015

Spectral Alignment of Networks

Soheil Feizi, Gerald Quon, Muriel Medard, Manolis Kellis, and Ali Jadbabaie

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Spectral Alignment of Networks

Soheil Feizi*, Gerald Quon*, Muriel Médard*, Manolis Kellis* and Ali Jadbabaie†

February 2015

Abstract

Network alignment refers to the problem of finding a bijective mapping across vertices of two or more graphs to maximize the number of overlapping edges and/or to minimize the number of mismatched interactions across networks. This paper introduces a network alignment algorithm inspired by eigenvector analysis which creates a simple relaxation for the underlying quadratic assignment problem. Our method relaxes binary assignment constraints along the leading eigenvector of an alignment matrix which captures the structure of matched and mismatched interactions across networks. Our proposed algorithm denoted by *EigeAlign* has two steps. First, it computes the Perron-Frobenius eigenvector of the alignment matrix. Second, it uses this eigenvector in a linear optimization framework of maximum weight bipartite matching to infer bijective mappings across vertices of two graphs. Unlike existing network alignment methods, EigenAlign considers both matched and mismatched interactions in its optimization and therefore, it is effective in aligning networks even with low similarity. We show that, when certain technical conditions hold, the relaxation given by EigenAlign is asymptotically exact over Erdös-Rényi graphs with high probability. Moreover, for modular network structures, we show that EigenAlign can be used to split the large quadratic assignment optimization into small subproblems, enabling the use of computationally expensive, but tight semidefinite relaxations over each subproblem. Through simulations, we show the effectiveness of the EigenAlign algorithm in aligning various network structures including Erdös-Rényi, power law, and stochastic block models, under different noise models. Finally, we apply EigenAlign to compare gene regulatory networks across human, fly and worm species which we infer by integrating genome-wide functional and physical genomics datasets from ENCODE and modENCODE consortia. EigenAlign infers conserved regulatory interactions across these species despite large evolutionary distances spanned. We find strong conservation of centrally-connected genes and some biological pathways, especially for human-fly comparisons.

Keywords. Network alignment, graph matching, graph isomorphism, matrix spectral theory, matrix perturbation, random graphs, gene regulatory networks.

1 Introduction

The term *network alignment* encompasses several distinct but related problem variants [1]. In general, network alignment aims to find a bijective mapping across two (or more) networks so that if two nodes are connected in one network, their images are also connected in the other network(s).

*Department of Electrical Engineering and Computer Science, MIT, Cambridge MA.

†Department of Electrical and Systems Engineering, University of Pennsylvania, and Sociotechnical Systems Research Center, MIT.

If such an errorless alignment scheme exists, network alignment is simplified to the problem of graph isomorphism [2]. However, in general, an errorless alignment scheme may not be feasible across two networks. In that case, network alignment aims to find a mapping with the minimum error and/or the maximum overlap.

Network alignment has a broad range of applications in different areas including biology, computer vision, and linguistics. For instance, network alignment has been used frequently as a comparative analysis tool in studying protein-protein interaction networks across different species [3–8]. In computer vision, network alignment has been used in image recognition by matching similar images [9, 10]; while it has been applied in ontology alignment to find relationships among different representations of a database [11, 12].

Finding an optimal alignment mapping across networks is computationally challenging [13], and is closely related to the quadratic assignment problem (QAP) [14]. However, owing to numerous applications of network alignment in different areas, several algorithms have been designed to solve this problem approximately. Some algorithms are based on linear [15], [16] or semidefinite [17, 18] relaxations of the underlying QAP, some methods use a Bayesian framework [19], or message passing [20], while other techniques use heuristics to find approximate solutions for the network alignment optimization [3, 4, 6]. We will review these methods in Section 2.2.

In general, existing network alignment methods have two major shortcomings. First, they only consider maximizing the number of overlapping edges (matches) across two networks and therefore, they ignore effects of mismatches (interactions that exist only in one of the networks). This can be critical in applications where networks have low similarity and therefore, there are many more expected possible mismatches than matches. Second, their performance is assessed mostly through simulations and/or validations with real data where an analytical performance characterization is lacked even in simple cases.

In this paper, we introduce a network alignment algorithm called *EigenAlign* which advances previous network alignment techniques in several aspects. EigenAlign creates a simple relaxation for the underlying QAP by relaxing binary assignment constraints linearly along the leading eigenvector of an alignment matrix which captures the structure of matched and mismatched interactions across networks. This leads to a solution for the underlying network alignment optimization which can be computed efficiently through an eigen decomposition step followed by a linear assignment step. Unlike existing network alignment methods, EigenAlign considers both matched and mismatched interactions in the optimization and therefore is effective in aligning networks even with low similarity. This is critical in comparative analysis of biological networks of distal species because there are numerous mismatched interactions across those networks, partially owing to extensive gene functional divergence due to processes such as gene duplication and loss.

EigenAlign advances existing network alignment methods in both algorithmic aspects, as well as qualitative aspects of the network alignment objective function, by considering both match and mismatch effects. Through analytical performance characterization, simulations on synthetic networks, and real-data analysis, we show that, the *combination* of these two aspects leads to an improved performance of the EigenAlign algorithm compared to existing network alignment methods in the literature. On simple examples, we isolate multiple aspects of the proposed algorithm and evaluate their individual contributions in the performance. We note that, existing network alignment packages may be improved by borrowing algorithmic and/or qualitative ideas of the proposed EigenAlign method. However, extending those methods is beyond the scope of this paper.

For an analytical characterization of the EigenAlign performance, we consider asymptotically

large Erdös-Rényi graphs [21] owing to their tractable spectral characterization. In particular, we prove that the EigenAlign solution is asymptotically optimal with high probability for Erdös-Rényi graphs, under some general conditions. Proofs are based on a characterization of eigenvectors of Erdös-Rényi graphs, along with a spectral perturbation analysis of the alignment matrix. Moreover, we evaluate the performance of the proposed method on real biological networks as well.

Although the EigenAlign relaxation leads to an efficient method to align large and complex networks which performs better than existent contenders, its relaxation may not be as tight as convex and semidefinite programming (SDP) relaxations of the underlying QAP that seek solutions in the intersection of orthogonal and stochastic matrices [17,18,22–25]. However, these methods have high computational complexity which prohibits their applications in aligning large networks. For modular network structures, we show that, EigenAlign can be used to split the large underlying quadratic assignment problem into small subproblems, enabling the use of computationally expensive SDP relaxations over each subproblem, in parallel. The key insight is that, the EigenAlign solution which can be computed efficiently even for large networks, provides a robust mapping of modules across networks. The resulting algorithm which we term *EigenAlign+SDP* is effective in aligning modular network structures with low computational complexity, even in high-noise levels.

We compare the performance of our proposed method against four existing network alignment methods based on belief propagation (NetAlign [20]), spectral decomposition (IsoRank [3]), Lagrange relaxation (Klau optimization [15]), and an SDP-based method [17] via simulations. Our simulation results illustrate the effectiveness of the EigenAlign algorithm in aligning various network structures including Erdös-Rényi, power law, and stochastic block structures, under different noise models. Moreover, we apply our method to compare gene regulatory networks across human, fly and worm species. First, we infer gene regulatory networks in these species by integrating genome-wide functional and physical datasets from ENCODE and modENCODE consortia, using both rank-based and likelihood-based integration approaches. We show that, inferred regulatory interactions have significant overlap with known interactions in TRANSFAC [26], REDfly [27] and EdgeDB [28] benchmarks, for human, fly and worm species, respectively, indicating the robustness and accuracy of the inference pipeline. Next, we apply the EigenAlign algorithm and other network alignment techniques to infer conserved regulatory interactions across these species using homolog gene mappings. Our results highlight the effectiveness of the EigenAlign algorithm in finding mappings which cause more matches and fewer mismatches across networks, compared to other network alignment techniques proposed in the literature. Using EigenAlign mappings, we find strong conservation of centrally-connected genes and some biological pathways, especially for human-fly comparisons.

The rest of the paper is organized as follows. In Section 2, we present the network alignment problem and review existent network alignment techniques. In Section 3, we introduce our proposed algorithm and discuss its relationship with the underlying quadratic assignment problem. Moreover, we present the optimality of our method over random graphs, under some general conditions. In Section 4, we consider the network alignment problem of modular networks and introduce an algorithm which solves it efficiently. In Section 5, we compare performance of our method with existent network alignment methods under different network structures and noise models. In Section 6, we introduce our network inference framework to construct integrative gene regulatory networks in different species. In Section 7, we illustrate applications of our method in comparative analysis of regulatory networks across species. In Section 8, we present optimality proofs of proposed methods. In Section 9, we conclude the paper and highlight future directions.

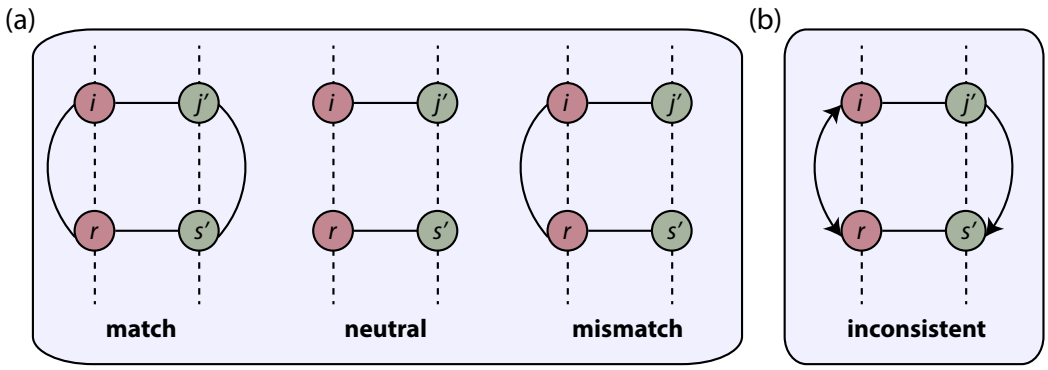

Figure 1: (a) An illustration of matched, mismatched, and neutral mappings, for undirected graphs. (b) An illustration of inconsistent mappings, for directed graphs, where they are matches in one direction, and mismatches in the other direction.

2 Network Alignment Problem Setup

2.1 Problem Formulation and Notation

In this section, we introduce the network alignment problem formulation. Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs (networks) where V_a and E_a represent set of nodes and edges of graph $a = 1, 2$, respectively. By abuse of notation, let G_1 and G_2 be their matrix representations as well where $G_a(i, j) = 1$ iff $(i, j) \in \mathcal{E}_a$, for $a = 1, 2$. Suppose network a has n_a nodes, i.e., $|V_a| = n_a$. We assume that networks are un-weighted (binary), and possibly directed. The proposed framework can be extended to the case of weighted graphs as well.

Let X be an $n_1 \times n_2$ binary matrix where $x(i, j') = 1$ means that node i in network 1 is mapped to node j' in network 2. The pair (i, j') is called a mapping edge across two networks. In the network alignment setup, each node in one network can be mapped to at most one node in the other network, i.e., $\sum_i x(i, j') \leq 1$ for all j', and similarly, $\sum_{j'} x(i, j') \leq 1$ for all i.

Let $\mathbf{y}$ be a vectorized version of X. That is, $\mathbf{y}$ is a vector of length $n_1 n_2$ 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 1 *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 1 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-b). These mappings are denoted as *inconsistent mappings*, defined as follows:

Definition 2 *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 $i \to r$, $r \to i$, and $j' \to s'$ exist, however, $s' \to j'$ does not exist, then mappings (i, j') and (r, s') are inconsistent.*

Existing network alignment formulations aim to find a mapping matrix X which maximizes the number of matches between networks. However, these formulations can lead to mappings which cause numerous mismatches, especially if networks have low similarity. In this paper, we propose a more general formulation for the network alignment problem which considers both matches and mismatches simultaneously.

For a given alignment matrix X across networks G_1 and G_2, we assign an *alignment score* by considering the number of matches, mismatches and neutrals caused by X:

$$\text{Alignment Score (X)} = s_1(\#\text{ of matches}) + s_2(\#\text{ of neutrals}) + s_3(\#\text{ of mismatches}), \tag{2.1}$$

where s_1, s_2, and s_3 are scores assigned to matches, neutrals, and mismatches, respectively. Note that, existing alignment methods ignore effects of mismatches and neutrals by assuming $s_2 = s_3 = 0$. In the following, we re-write (2.1) as a quadratic assignment formulation.

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

Definition 3 *Let $\{i, r\} \subseteq V_1$ and $\{j', s'\} \subseteq V_2$, where $x(i, j') = 1$ and $x(r, s') = 1$.*

$$A\big[(i, j'), (r, s')\big] = \begin{cases} s_1, & \text{if } (i, j') \text{ and } (r, s') \text{ are matches,} \\ s_2, & \text{if } (i, j') \text{ and } (r, s') \text{ are neutrals,} \\ s_3, & \text{if } (i, j') \text{ and } (r, s') \text{ are mis-matches,} \end{cases} \tag{2.2}$$

where s_1, s_2, and s_3 are scores assigned to matches, neutrals, and mismatches, respectively.

We can re-write (2.2) as follows:

$$A\big[(i, j'), (r, s')\big] = (s_1 + s_2 - 2s_3)G_1(i, r)G_2(j', s') + (s_3 - s_2)(G_1(i, r) + G_2(j', s')) + s_2. \tag{2.3}$$

We can summarize (2.2) and (2.3) as follows:

$$A = (s_1 + s_2 - 2s_3)(G_1 \otimes G_2) + (s_3 - s_2)(G_1 \otimes \mathbb{1}_{n_2}) + (s_3 - s_2)(\mathbb{1}_{n_1} \otimes G_2) + s_2(\mathbb{1}_{n_1} \otimes \mathbb{1}_{n_2}), \tag{2.4}$$

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

Remark 1 A similar scoring scheme can be used for directed graphs. However when graphs are directed, some mappings can be inconsistent according to Definition 2, 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 network edges are

distributed according to a Bernoulli distribution with parameter p. Then, the probability of having an inconsistent mapping across networks is equal to $4p^3(1-p)$. Therefore, their effect in network alignment is negligible specially for large sparse networks. Throughout the paper, for directed graphs, we assume inconsistent mappings have negligible effect unless it is mentioned explicitly.

Alignment scores s_1, s_2 and s_3 of (2.2) can be arbitrary in general. However, in this paper we consider the case where $s_1 > s_2 > s_3 > 0$ with the following rationale: Suppose a mapping matrix X has a total of κ mapping edges. For example, if networks have $n_1 = n_2 = n$ nodes and there is no unmapped nodes across two networks, $\kappa = n$. The total number of matches, mismatches and neutrals caused by this mapping is equal to $\binom{\kappa}{2}$. Thus, for mapping matrices with the same number of mapping edges, without loss of generality, one can assume that, alignment scores are strictly positive $s_1, s_2, s_3 > 0$ (otherwise, a constant can be added to the right-hand side of (2.2)). In general, mappings with high alignment scores might have slightly different number of mapping edges owing to unmapped nodes across the networks which has a negligible effect in practice. Moreover, in the alignment scheme, we wish to encourage matches and penalize mismatches. Thus, throughout this paper, we assume $s_1 > s_2 > s_3 > 0$.

Remark 2 In practice, some mappings across two networks may not be possible owing to additional side information. The set of possible mappings across two networks 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 network alignment is called *unrestricted*. However, if some mappings across two networks are prevented (i.e., $y_{i,j'} = 0$, for $(i, j') \notin \mathcal{R}$), then the problem of network alignment is called *restricted*.

In the following, we present the network alignment optimization which we consider in this paper:

Definition 4 (Network Alignment Problem Setup) *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two binary networks. Network alignment aims to solve the following optimization:*

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T A \mathbf{y}, \\ & \sum_i y_{i,j'} \le 1, \quad \forall i \in V_1, \\ & \sum_{j'} y_{i,j'} \le 1, \quad \forall j' \in V_2, \\ & y_{i,j'} \in \{0,1\}, \quad \forall (i,j') \in V_1 \times V_2, \\ & y_{i,j'} = 0, \quad \forall (i,j') \notin \mathcal{R}, \end{aligned} \tag{2.5}$$

where A is defined according to (2.3), and $\mathcal{R} \subseteq V_1 \times V_2$ is the set of possible mappings across two networks.

In the following, we re-write (2.5) using the trace formulation of a standard QAP. Here, we consider undirected networks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with n_1 and n_2 nodes, respectively. Without loss of generality, we assume $n_1 \le n_2$. Moreover, we assume that all nodes of the network G_1 are mapped to nodes of the network G_2.

Define $\xi_1 \triangleq s_1 + s_2 - 2s_3$, $\xi_2 \triangleq s_3 - s_2$ and $\xi_3 \triangleq s_2$. We can rewrite the objective function of Optimization (2.5) as follows:

$$
\begin{aligned}
\mathbf{y}^T A\mathbf{y} &= \xi_1 \mathbf{y}^T (G_1 \otimes G_2)\mathbf{y} + \xi_2 \mathbf{y}^T (G_1 \otimes \mathbb{1}_{n_2})\mathbf{y} + \xi_2 \mathbf{y}^T (\mathbb{1}_{n1} \otimes G_2)\mathbf{y} + \xi_3 \mathbf{y}^T (\mathbb{1}_{n_1} \otimes \mathbb{1}_{n_2})\mathbf{y} \\
&= \xi_1 Tr(G_1 X G_2 X^T) + \xi_2 Tr(G_1 X \mathbb{1}_{n_2} X^T) + \xi_2 Tr(\mathbb{1}_{n_1} X G_2 X^T) + \xi_3 Tr(\mathbb{1}_{n_1} X \mathbb{1}_{n_2} X^T) \\
&= Tr\big((G_1 + \frac{\xi_2}{\xi_1})X(\xi_1 G_2 + \xi_2)X^T\big) + (\xi_3 - \frac{\xi_2^2}{\xi_1}) Tr(\mathbb{1}_{n_1} X \mathbb{1}_{n_2} X^T) \\
&= Tr(G_1' X G_2' X^T) + (\xi_3 - \frac{\xi_2^2}{\xi_1})(\min(n_1, n_2))^2 \\
&= Tr(G_1' X G_2' X^T) + constant,
\end{aligned}
$$

where

$$
\begin{aligned}
G_1' &\triangleq G_1 + \frac{\xi_2}{\xi_1} = G_1 + \frac{s_3 - s_2}{s_1 + s_2 - 2s_3}, \\
G_2' &\triangleq \xi_1 G_2 + \xi_2 = (s_1 + s_2 - 2s_3)G_2 + (s_3 - s_2).
\end{aligned}
$$

Thus, the network alignment optimization (2.5) can be reformulated as follows:

$$
\begin{aligned}
\max_X \quad & Tr(G_1' X G_2' X^T), \\
& \sum_i x_{i,j'} \le 1, \quad \forall i \in V_1, \\
& \sum_{j'} x_{i,j'} \le 1, \quad \forall j' \in V_2, \\
& x_{i,j'} \in \{0,1\}, \quad \forall (i,j') \in V_1 \times V_2, \\
& x_{i,j'} = 0, \qquad \forall (i,j') \notin \mathcal{R}.
\end{aligned} \tag{2.6}
$$

Remark 3 Some network alignment formulations aim to align paths [7] or subgraphs [8,29] across two (or multiple) networks. The objective of these methods is different than the one of our network alignment optimization (2.5), where a bijective mapping across nodes of two networks is desired. However, obtained solutions of these different methods may be related. For instance, a bijective mapping across nodes of two networks can provide information about conserved pathways and/or subgraphs across networks, and vice versa.

2.2 Prior Work

Network alignment optimization (2.5) is an example of a quadratic assignment problem (QAP) [14]. Reference [30] shows that approximating a solution of maximum quadratic assignment problem within a factor better than $2^{log^{1-\epsilon} n}$ is not feasible in polynomial time in general. However, owing to various applications of QAP in different areas, several works have attempted to solve it approximately. 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 [14,31].

- **Exact search methods:** these methods provide a global optimal solution for the quadratic assignment problem. However, owing to their high computational complexity, they can only

be applied to very small problem instances. Examples of exact algorithms include methods based on branch-and-bound [32] and cutting plane [33].

- **Linearizations:** these methods attempt to solve QAP by eliminating the quadratic term in the objective function of Optimization (2.5), 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 [34], Kaufmann and Broeckx linearization [35], Frieze and Yadegar linearization [36], and Adams and Johnson linearization [37]. These linearizations can provide bounds on the optimal value of the underlying QAP [30]. 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.

- **Semidefinite/convex relaxations and bounds:** these methods aim to compute a bound on the optimal value of the network alignment optimization, by considering the alignment matrix in the intersection of the sets of orthogonal and stochastic matrices. The provided solution by these methods may not be a feasible solution of the original quadratic assignment problem. Examples of these methods include orthogonal relaxations [22], projected eigenvalue bounds [23], convex relaxations [18,24,25], and matrix splittings [17]. In the computer vision literature, [38,39] use spectral techniques to approximately solve QAP by inferring a cluster of assignments over the feature network. Then, they use a greedy approach to reject assignments with low associations.

  In particular, [17] introduces a convex relaxation of the underlying network alignment optimization based on matrix splitting which provides bounds on the optimal value of the underlying QAP. The proposed 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 networks, limiting its applicability to alignment of large real networks [1]. In Section 4, we address these issues and introduce a hybrid method based on our proposed scheme in Section 3, and the semidefinite relaxation of [17] to align large modular network structures with low computational complexity.

- **Other methods:** there are several other techniques to approximately solve network alignment optimization. Some methods use a Lagrangian relaxation [15], Bayesian framework [19], or message passing [20], or some other heuristics [3,4,6]. In Section 5, we assess the performance of some of these network alignment techniques through simulations.

Besides described method-specific limitations of existing network alignment methods, these methods have two general shortcomings: First, they only consider maximizing the number of matches and therefore ignore effects of mismatches across networks (i.e., they assume $s_2 = s_3 = 0$ in (2.5)). This can be critical in applications in which networks show low similarity (i.e., there are much more expected possible mismatches than matches). Second, their performance assessment is mostly based on simulations and/or validations with real data without an analytical performance characterization. In this paper, we propose a network alignment algorithm which considers both matches and mismatches in the alignment scheme. Moreover, we analyze its performance over

[1] In our simulations, networks should have approximately less than 70 nodes to be able to run it on an ordinary laptop.

random graphs and show that, the proposed relaxation is asymptotically exact under some technical conditions.

2.3 Network Alignment and Graph Isomorphism

Network alignment optimization (2.5) is closely related to the problem of *graph isomorphism*, defined as follows:

Definition 5 (Graph Isomorphism) *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two binary networks. 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 networks G_1 and G_2, the problem of graph isomorphism aims to find the permutation matrix P such that $G_1 = PG_2P^T$. The graph isomorphism problem seems computationally intractable in general. However, its computational complexity is still unknown [13].

Problems of network alignment and graph isomorphism are related to each other. Loosely speaking, network alignment aims to minimize the distance between premuted versions of two networks (or, alternatively to maximize their overlap). Therefore, if the underlying networks are isomorphic, an optimal solution of the network alignment optimization should be the same (or close) to the underlying permutation matrix P, where $G_1 = PG_2P^T$. In the following lemma, we formalize such a connection between the network alignment optimization and the classical graph isomorphism problem:

Lemma 1 *Let G_1 and G_2 be two isomorphic Erdös-Rényi graphs [21] 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 network alignment objective function of Optimization (2.5).*

Proof Let A be the alignment network of graphs G_1 and G_2. 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,

$$\begin{aligned}
\frac{1}{n^2}\mathbb{E}[\tilde{\mathbf{y}}_2^T A\tilde{\mathbf{y}}_2] =& (1-\rho)\big[ps_1 + (1-p)s_2\big] + \rho\big[p^2 s_1 + (1-p)^2 s_2 + 2p(1-p)s_3\big] \\
<& (1-\rho)\big[ps_1 + (1-p)s_2\big] + \rho\big[p^2 s_1 + (1-p)^2 s_2 + p(1-p)(s_1+s_2)\big] \\
=& (1-\rho)\big[ps_1 + (1-p)s_2\big] + \rho\big[ps_1 + (1-p)s_2\big] \\
=& ps_1 + (1-p)s_2 \\
=& \frac{1}{n^2}\mathbb{E}[\mathbf{y}_1^T A\mathbf{y}_1].
\end{aligned} \tag{2.7}$$

■

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 Erdös-Rényi graphs 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 network alignment objective function of Optimization (2.5).*

Proof Similarly to the proof of Lemma 1, let A be the alignment network of graphs 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:

$$\begin{aligned} a' \triangleq & p(1-q)s_1 + (1-p)(1-q)s_2 + (pq + (1-p)q)s_3, \\ b' \triangleq & \big(p^2(1-q) + pq(1-p)\big)s_1 \\ & + \big((1-p)^2(1-q) + pq(1-p)\big)s_2 \\ & + \big(2p(1-p)(1-q) + 2p^2q\big)s_3. \end{aligned} \tag{2.8}$$

Thus,

$$a' - b' = p(1-p)(1-2q)(s_1 + s_2 - 2s_3) + q(1-2p)s_3. \tag{2.9}$$

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 (2.9), $a' > b'$. Thus we have,

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

■

Finding an isomorphic mapping across sufficiently large Erdös-Rényi graphs can be done efficiently with high probability (w.h.p.) through canonical labeling [40]. Canonical labeling of a network 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 [41]. One example of canonical labeling is the degree neighborhood of a vertex defined as a sorted list of neighborhood degrees of vertices [40]. Note that, network alignment formulation is more general than the one of graph isomorphism; network alignment aims to find an optimal mappings across two networks which are not necessarily isomorphic.

Remark 4 In [42], Babai, Erdös, and Selkow derive an interesting and perhaps a counter-intuitive result on random graph isomorphism. Based on their result, *any two infinite random graphs are isomorphic with probability one*. For instance, consider two infinite Erdös-Rényi graphs G_1 and G_2 where $Pr[G_1(i,j) = 1] = p_1$ and $Pr[G_2(i,j) = 1] = p_2$ and $p_1, p_2 \neq 0, 1$. The result of [42] indicates that, G_1 and G_2 are isomorphic even if $p_1 \neq p_2$. This may seem counter-intuitive as two networks may seem to have different densities. However, this result is only true for *infinite graphs*, not asymptotically large ones. The difference may seem subtle but significant. In infinite graphs, notions of graph size, graph density, etc. are different than the ones for finite graphs. Throughout

this paper, we only consider finite or asymptotically large graphs, not infinite ones. In the following, we give some intuition on the results of [42] about infinite graphs.

The proof of the result of [42] is based on a notion for infinite graphs called *the extension property* which roughly states that, an infinite graph G has the extension property if for any two disjoint finite subsets of nodes V_1 and V_2, there exists a vertex $v \in V - V_1 - V_2$ such that v is connected to all vertices in V_1 and to no vertices in V_2. If two infinite graphs have extension properties, they are isomorphic with probability one. One way to prove this is to construct an infinite sequences of mappings by considering disjoint subsets of nodes V_1 and V_2, and adding a node v according to the extension property.

Finally, it is straightforward to show that an infinite Erdös-Rényi graph with parameter $p \neq 0, 1$ has the extension property with probability one. This is analogous to the monkey-text book problem: if a monkey randomly types infinite letters, with probability one, he will type any given text book. This is because the number of letters in a text book is finite and therefore, the probability that the letter sequence of a given text book does not appear in a random infinite letter sequence is zero. Now consider two finite disjoint subsets V_1 and V_2 over an infinite graph. Note that being finite is key here. Similarly, the probability that a vertex v exists such that it is connected to all vertices in V_1 and to no vertices in V_2 is one because its complement set has zero probability (or it has zero Lebesgue measure if we map binary sequences to numbers in $[0, 1]$.).

3 EigenAlign Algorithm

In this section, we introduce *EigenAlign*, an algorithm which solves a relaxation of the network alignment optimization (2.5) leveraging spectral properties of networks. Unlike other alignment methods, EigenAlign considers both matches and mismatches in the alignment scheme. Moreover, we prove its optimality (in an asymptotic sense) in aligning Erdös-Rényi graphs under some technical conditions. In the following, we describe the EigenAlign algorithm:

Algorithm 1 (EigenAlign Algorithm) *Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two binary networks whose corresponding alignment network is denoted by A according to (2.3). EigenAlign algorithm solves the network alignment optimization (2.5) in two steps:*

- **Eigen Decomposition Step:** *In this step, we compute $\mathbf{v}$, an eigenvector of the alignment network A with the maximum eigenvalue.*
- **Linear Assignment Step:** *In this step, we solve the following maximum weight bipartite matching optimization:*

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{v}^T \mathbf{y}, \\ & \sum_{j'} y_{i,j'} \leq 1, \quad \forall i \in V_1, \\ & \sum_{i} y_{i,j'} \leq 1, \quad \forall j' \in V_2, \\ & y_{i,j'} \in \{0, 1\}, \quad \forall (i, j') \in V_1 \times V_2, \\ & y_{i,j'} = 0, \qquad \forall (i, j') \notin \mathcal{R}. \end{aligned} \tag{3.1}$$

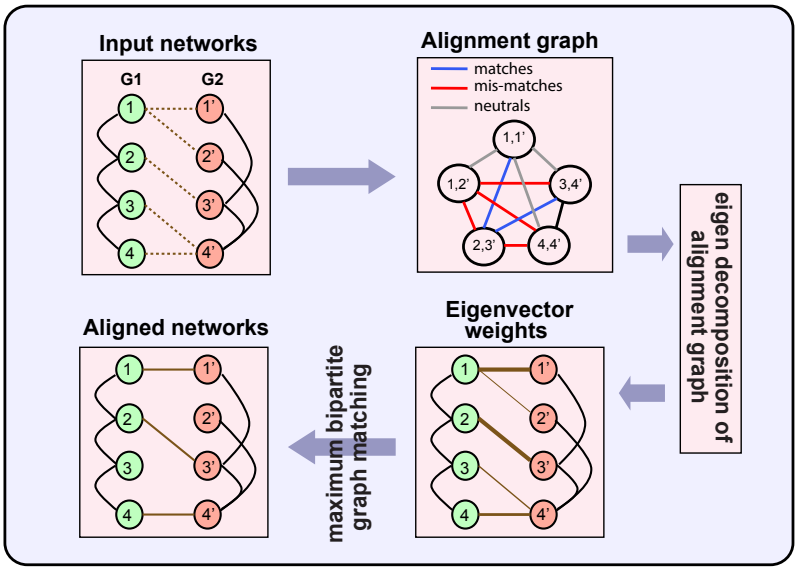

Figure 2: Framework of EigenAlign algorithm 1.

This framework is depicted in Figure 2. In the rest of this section, we provide intuition on different steps of the EigenAlign algorithm through both quadratic assignment relaxation argument as well as a fixed point analysis. In Section 3.3, we discuss optimality of EigenAlign over random graphs.

3.1 EigenAlign as Relaxation of Quadratic Assignment

In this section, we explain EigenAlign as a relaxation of the underlying quadratic assignment optimization (2.5). For simplicity, we assume all mappings across networks are possible (i.e., $\mathcal{R} = \{(i, j') : \forall i \in V_1, \forall j' \in V_2\}$). In the restricted network alignment setup, without loss of generality, one can eleminate 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. By these assumptions, Optimization (2.5) can be simplified to the following optimization:

$$\begin{aligned} \max_{\mathbf{y}} \quad & \mathbf{y}^T A \mathbf{y}, \\ & y_{i,j'} \in \{0, 1\}, \quad \forall (i, j') \in V_1 \times V_2. \end{aligned} \tag{3.2}$$

To approximate a solution of this optimization, we relax integer constraints to constraints over a hyper-sphere restricted by hyper-planes (i.e., $\|\mathbf{y}\|_2 \leq 1$ and $\mathbf{y} \geq 0$). Using this relaxation, Optimization (3.2) is simplified to the following optimization:

$$\max_{\mathbf{y}} \quad \mathbf{y}^T A \mathbf{y}, \tag{3.3}$$
$$\|\mathbf{y}\|_2 \leq 1,$$
$$\mathbf{y} \geq 0.$$

In the following, we show that $\mathbf{v}$, the leading eigenvector of the alignment matrix A is an optimal solution of Optimization (3.3).

Suppose $\mathbf{y}_1$ is an optimal solution of Optimization (3.3). Let $\mathbf{y}_2$ be a solution of the following optimization which ignores non-negativity constraints:

$$\max_{\mathbf{y}} \quad \mathbf{y}^T A \mathbf{y}, \tag{3.4}$$
$$\|\mathbf{y}\|_2 \leq 1.$$

Following the Rayleigh – Ritz formula, the leading eigenvector of the alignment matrix is an optimal solution of Optimization (3.4) (i.e., $\mathbf{y}_2 = \mathbf{v}$). Now we use the following theorem to show that in fact $\mathbf{y}_1 = \mathbf{v}_1$:

Theorem 1 (Perron–Frobenius Theorem [2]) *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.*

Since $\mathbf{y}_2$ is a solution of a relaxed version of Optimization (3.4), we have $\mathbf{y}_2^T A \mathbf{y}_2 \geq \mathbf{y}_1^T A \mathbf{y}_1$. Using this inequality along with Perron-Frobenius Theorem lead to $\mathbf{y}_1 = \mathbf{v}$, as the unique solution of optimization (3.3).

The solution of the eigen decomposition step assigns weights to all possible mappings across networks ignoring bijective constraints. However, in the network alignment setup, each node in one network can be mapped to at most one node in the other network. To satisfy these constraints, we use eigenvector weights in a linear optimization framework of maximum weight bipartite matching setup of Optimization (3.1) [43].

Remark 5 Many existing network alignment techniques are based on iterative algorithms [3,4,20]. The EigenAlign relaxation of the underlying quadratic assignment problem can be viewed as the following fixed point solution using a linear mapping function which can be solved iteratively:

$$y(t_1) = \frac{1}{\lambda} \sum_{t_2} A(t_1, t_2) y(t_2), \tag{3.5}$$

where λ is a constant and t_i represents a node in the alignment network A. Note that, this mapping function is not necessarily a contraction. Instead of using an iterative approach, EigenAlign solves this relaxation in a closed form using the leading eigenvector of the alignment network. In particular, (3.5) can be re-written as,

$$A\mathbf{y} = \lambda \mathbf{y}, \tag{3.6}$$

where $\mathbf{y}$ is the vector of mapping weights whose elements represent a measure of centrality of nodes in the alignment network. If A is diagonalizable, eigenvectors of matrix A provide solutions for this equation, where λ is the eigenvalue corresponding to eigenvector $\mathbf{y}$. However, eigenvectors of matrix A can have negative components which are not allowed in the mapping function since $y_{i,j'} \geq 0$. Since components of matrix A are all positive, similarly to discussions of (3.3), according to Perron-Frobenius Theorem 1, all components of the eigenvector associated with the maximum positive eigenvalue are positive. This result holds for non-negative matrices as well if they are strongly connected. Therefore, the leading eigenvector of matrix A satisfies desired conditions for the mapping function and provides alignment scores for the nodes in the alignment network.

Remark 6 Suppose G_1 and G_2 are two undirected graphs with n_1 and n_2 nodes, respectively. IsoRank [3] is a network alignment method which aligns networks using neighborhood similarities of nodes. Let I be the identity matrix of size $n_1 \times n_1$. Suppose D_1 and D_2 are diagonal matrices of sizes $n_1 \times n_1$ and $n_2 \times n_2$ whose diagonal elements are node degrees of networks G_1 and G_2, respectively. The first step of the IsoRank algorithm solves the following fixed point equation:

$$\mathbf{y} = (I \otimes D_1^{-1})A(I \otimes D_2^{-1})\mathbf{y}, \tag{3.7}$$

where $\otimes$ represents matrix Kronecker product, and $s_2 = s_3 = 0$ in (2.3). IsoRank solves (3.7) using a power iteration method. The IsoRank algorithm has several differences with EigenAlign in both algorithmic and qualitative aspects. In the qualitative aspect, IsoRank ignores effects of mismatches across networks by assuming $s_2 = s_3 = 0$. In algorithmic aspects, IsoRank uses a stationary centrality measure in the alignment network, having a degree scaling step which is based on the assumption of uniformly random distribution of alignment scores over potential mappings. Moreover, it uses a greedy method to select bijective mappings across networks, instead of the linear optimization framework used in the EigenAlign algorithm. In Sections 5 and 7, over both synthetic and real networks, we show that, the EigenAlign solution has significantly better performance compared to the one of IsoRank. Moreover, in Section 5, we isolate multiple aspects of the EigenAlign algorithm on some toy examples, and evaluate their individual contributions in its performance. We note that, existent network alignment methods such as IsoRank may be improved by borrowing algorithmic and/or qualitative ideas of the proposed EigenAlign method. However, these extensions are beyond the scope of this paper.

3.2 Computational Complexity of EigenAlign

In this part, we analyze computational complexity of the EigenAlign Algorithm 1. Suppose the number of nodes of networks G_1 and G_2 are in the order of $\mathcal{O}(n)$. Let $k = \mathcal{O}(|\mathcal{R}|)$ be the number of possible mappings across two networks. In an unrestricted network alignment setup, we may have $k = \mathcal{O}(n^2)$. However, in sparse network alignment applications, $k = \mathcal{O}(n)$. EigenAlign has three steps:

- (i) First, the alignment network A should be formed which has a computational complexity of $\mathcal{O}(k^2)$ because all pairs of possible mappings should be considered.
- (ii) In the eigen decomposition step, we need to compute the leading eigenvector of the alignment network. This operation can be performed in almost linear time in k using QR algorithms and/or power methods [44]. Therefore, the worst case computational complexity of this part is $\mathcal{O}(k)$.

- (iii) Finally, we use eigenvector weights in a maximum weight bipartite matching algorithm which can be solved efficiently using linear programming or Hungarian algorithm [43]. 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 network are allowed to be mapped to small subsets of nodes in the other network), this cost can be reduced significantly. In Section 7, we see this structure in aligning regulatory networks across species as genes are allowed to be aligned to homologous genes within their gene families.

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

Remark 7 For large networks, to reduce the overall computational complexity, the linear assignment optimization may be replaced by a greedy bipartite matching algorithm. 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 $\mathcal{O}(k\log(k) + nk)$.

3.3 Performance Characterization of EigenAlign Over Erdös-Rényi Graphs

In this section, we analyze optimality of the EigenAlign algorithm over Erdös-Rényi graphs, for both isomorphic and non-isomorphic cases, and under two different noise models. In this section, we only consider finite and asymptotically large graphs. For arguments on infinite graphs, see Section 2.3.

Suppose $G_1 = (V_1, E_1)$ is an undirected Erdös-Rényi graph with n nodes where $Pr[G_1(i,j) = 1] = p$. Self-loops are allowed as well. 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,

$$\tilde{G}_1 = G_1 \odot (1 - Q) + (1 - G_1) \odot Q, \tag{3.8}$$

  where $\odot$ represents the Hadamard product, and Q is a binary 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 $G_1 \odot (1 - Q) + (1 - G_1) \odot Q$ flips edges of G_1 independently randomly with probability p_e.

- Noise Model II: In this model, we have,

$$\tilde{G}_1 = G_1 \odot (1 - Q) + (1 - G_1) \odot Q', \tag{3.9}$$

  where Q and Q' are binary 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 independently 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 Erdös-Rényi graph with parameter p, choosing

$$p_{e_2} = \frac{pp_e}{1 - p}, \tag{3.10}$$

  leads to the expected density of networks G_1 and G_2 be p.

We define G_2 as follows:

$$G_2 = P\tilde{G}_1 P^T, \tag{3.11}$$

where P is a permutation matrix. Throughout this section, we assume that, we are in the restricted network alignment regime: we desire to choose n mappings $i \leftrightarrow j'$ across two networks among $|\mathcal{R}| = kn$ possible mappings where $i \in V_1$, $j' \in V_2$, and $k > 1$. n true mappings ($i \leftrightarrow i'$ if $P = I$) are included in $\mathcal{R}$, while the remaining $(k-1)n$ mappings are selected independently randomly. 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 $s_1 > s_2 > s_3 > 0$.

Theorem 2 (EigenAlign over Erdös-Rényi graphs) *Let $\tilde{P}$ be the solution of the EigenAlign Algorithm 1. Then, under both noise models (3.8) and (3.9), if $0 < p < 1/2$, and $0 \le p_e < 1/2$, then as $n \to \infty$, the error probability goes to zero:*

$$Pr\Big[\frac{1}{n}\|\tilde{P} - P\|\Big] \to 0.$$

Theorem 2 states that, the EigenAlign algorithm is able to recover the underlying permutation matrix which relates networks G_1 and G_2 to each other according to (3.11). On the other hand, according to Lemma 2, this permutation matrix is in fact optimizes the expected network alignment score.

Proposition 2 *Under conditions of Theorem 2, the permutation matrix inferred by EigenAlign maximizes the expected network alignment objective function defined according to Optimization (2.5).*

In noise models (3.8) and (3.9), 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 (EigenAlign on Isomorphic Erdös-Rényi graphs) *Let G_1 and G_2 be two isomorphic Erdös-Rényi graphs with n nodes such that $G_1 = PG_2P^T$, where P is a permutation matrix. Under conditions of Theorem 2, as $n \to \infty$, the error probability of EigenAlign solution goes to zero.*

We present proofs of Theorem 2 and Corollary 1 in Sections 8.1 and 8.2. In the following, we sketch main ideas of their proofs:

Since input networks G_1 and G_2 are random graphs, the alignment network formed according to (2.3) will be a random graph as well. The first part of the proof is to characterize the leading eigenvector of this random alignment network. To do this, we first characterize the leading eigenvector of the expected alignment network which in fact is a deterministic graph. In particular, in Lemma 3, we prove that, eigenvector scores assigned to true mappings is strictly larger than the ones assigned to false mappings. To prove this, we characterize top eigenvalues and eigenvectors of the expected alignment network algebraically. The restricted alignment condition (i.e., $|\mathcal{R}| = kn$) is necessary to have this bound. Then, we use Wedin Sin Theorem 8 from perturbation theory, Gershgorian circle

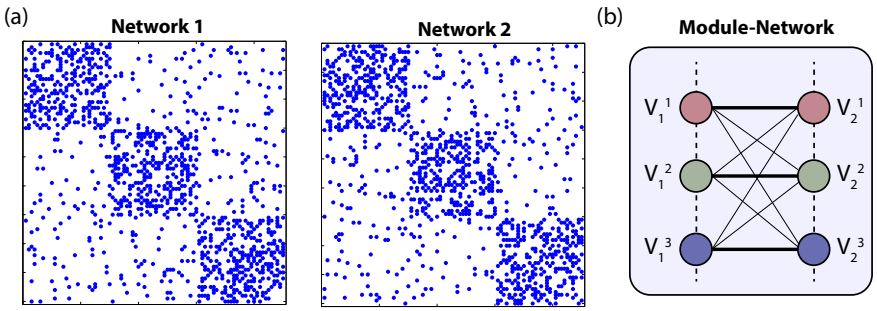

Figure 3: (a) Example modular network structures of Definition 6. (b) An illustration of the module-network bipartite graph whose nodes correspond to modules in the original networks.

Theorem 7 from spectral matrix theory, and Chernoff bound to characterize the leading eigenvector of the random alignment network for sufficiently large n. Finally, we use Chebyshev's inequality to show that the error probability of the EigenAlign algorithm is asymptotically zero w.h.p.

Remark 8 Finding an isomorphic mapping across asymptotically large Erdös-Rényi graphs (Corollary 1) is a well studied problem and can be solved efficiently through canonical labeling [40]. However, those techniques do not address a more general network alignment problem similar to the setup considered in Theorem 2. For more details, see Section 2.3.

Remark 9 Theorem 2 and Corollary 1 consider a restricted network alignment case where $|\mathcal{R}| = kn$. As explained briefly in the proof sketch and with more details in Lemma 3, this technical condition is necessary to show that, expected eigenvector scores of true mappings are strictly larger than the ones of false mappings as $n \to \infty$. In Section 5 and through simulations, we show that, error of the EigenAlign algorithm is empirically small even in an unrestricted network alignment setup.

4 Alignment of Modular Networks

As we discussed in Section 3.1, EigenAlign provides a simple relaxation of the QAP in (2.5), based on the eigenvector centrality of the alignment mappings. This can be thought of as a linearization of the quadratic assignment cost along the direction of the eigenevctor centrality. EigenAlign relaxes integer constraints to constraints over a hyper-sphere restricted by hyper-planes. This relaxation leads to an efficient method to align large networks. There are several methods based on convex relaxations of the underlying QAP that seek solutions in the intersection of orthogonal and stochastic matrices [17, 18, 22–25]. In general, these relaxations are tighter than the one used in the EigenAlign algorithm and lead to a better approximation of the optimal solution. However, these methods have high computational complexity which limits their applicability in aligning large networks. For instance, as reported in [17] and also according to our experiments, to be able to run the SDP-based method proposed in [17] on an ordinary laptop, networks should not have more than around 70 nodes. Although this threshold can be improved by a better implementation of the method, the SDP-based method is not scalable for large networks. Our key idea is to use the EigenAlign solution to make the SDP-based methods scalable for modular network structures.

We therefore consider the network alignment problem of modular network structures. We propose a method, which we term *EigenAlign+SDP*, that uses the EigenAlign algorithm along with an SDP-based relaxation of the underlying QAP. The proposed algorithm uses the EigenAlign solution to identify small subgraphs (modules, or blocks) across two networks which are likely to be aligned to each other. Then, it uses a more computationally expensive SDP-based method to solve multiple small-size network alignment problems, in parallel. The proposed method is based on this key insight that, the EigenAlign solution provides a robust mapping of modules across networks, which enables the use of more expensive and accurate optimization methods over small sub-problems.

In this section, we consider stochastic block network structures:

Definition 6 *Let $G_1 = (V_1, E_1)$ be an undirected graph. Suppose there is a partition of nodes $\{V_1^1, V_1^2, \ldots, V_1^m\}$, where nodes in the same partition are connected to each other independently randomly with probability p, while nodes across partitions are connected to each other independently randomly with probability q.*

Let $\tilde{G}_1$ be a noisy version of G_1. Here, we consider the noise model II (3.9) as it is more general, while all arguments can be extended to the noise model I (3.8). We assume G_2 is a permuted version of $\tilde{G}_1$ according to (3.11), i.e., $G_2 = P\tilde{G}_1 P^T$. Figure 3-a demonstrates example networks according to this model.

In the following, we present the *EigenAlign+SDP* method which aims to infer an optimal alignment across G_1 and G_2.

Algorithm 2 (EigenAlign+SDP Algorithm) *Let $G_1 = (V_1, E_1)$ be a stochastic block matrix of Definition 6, and $G_2 = (V_2, E_2)$ is defined according to (3.11). The EigenAlign+SDP algorithm solves the network alignment optimization (2.5) in the following steps:*

- **EigenAlign Step:** *In this step, we compute the EigenAlign solution across G_1 and G_2.*
- **Spectral Partitioning Step***: In this step, we use a spectral partitioning method [45] to partition each network to m blocks.*
- **Module Alignment Step***: In this step, we use the EigenAlign solution in a maximum weight bipartite matching optimization to compute an optimal alignment of modules across G_1 and G_2.*
- **SDP-based Alignment Step***: In this step, we use a SDP-based relaxation of the underlying QAP [17] followed by a maximum weight bipartite matching step, to compute node alignments in each module pairs.*

In the following, we explain the above steps with more details. In the EigenAlign step, we use EigenAlign algorithm 1 to find a mapping across networks G_1 and G_2, denoted by X^*. In the spectral partitioning step, we use the algorithm proposed in [45] to find m blocks (modules) in each network. These modules are denoted by $\{V_1^1, V_1^2, \ldots, V_1^m\}$ and $\{V_2^1, V_2^2, \ldots, V_2^m\}$, in networks G_1 and G_2, respectively. Note that other network clustering methods can be used in this step alternatively. Moreover, here we assume that, the number of clusters (blocks) m is known. Otherwise, it can be learned using Bayesian [46] or Monte Carlo [47] techniques. In the module

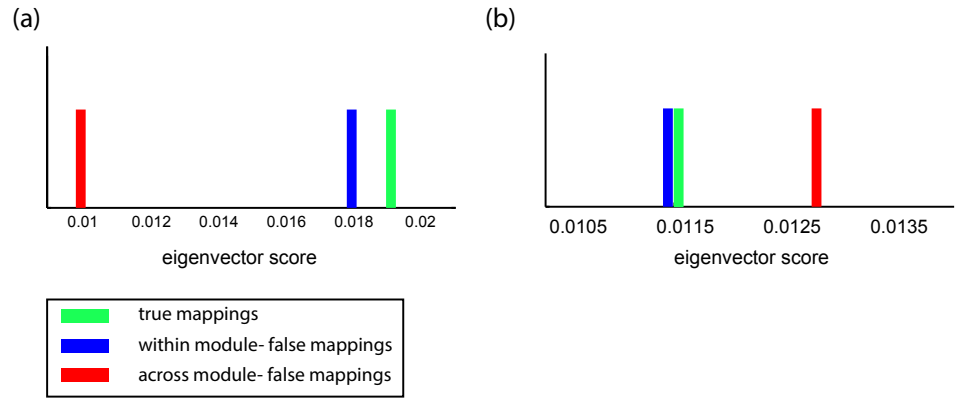

Figure 4: Eigenvector scores of within and across module mappings of the expected alignment network with parameters $n = 20$, $m = 4$, $p = 0.3$, $q = 0.01$, and (a) $\alpha = 100$, (b) $\alpha = 10$. Parameters of panel (a) satisfy conditions of Theorem 3.

alignment step, we use the EigenAlign solution X^* to infer a mapping across inferred modules. To do this, we form a bipartite module-network whose nodes in the first and second layers correspond to $\{V_1^1, V_1^2, \ldots, V_1^m\}$, and $\{V_2^1, V_2^2, \ldots, V_2^m\}$, respectively (Figure 3-b). The edge weight connecting node i in the first layer of this graph to node j' in the second layer (i.e., $w(i, j')$) is computed as follows:

$$w(i, j') = \sum_{\substack{a \in V_1^i \\ b \in V_2^j}} X^*(a, b). \tag{4.1}$$

We use these weights in a maximum bipartite matching optimization to infer a one-to-one mapping across modules of two networks. Note that, other methods based on random walks over aggregation of Markov chains [48] can be used in this step to find module mappings. In the last step, we use an SDP-based relaxation of the underlying QAP [17] along with a linear programming step of maximum weight bipartite matching to compute node alignments in each module-pair. Note that, in this step, other methods based on convex/semidefinite relaxations of the underlying QAP can be used as well.

In the following, we prove that the EigenAlign solution provides a robust module-level mapping across networks even in the high-noise regime.

Define,

$$\begin{aligned} \mathcal{M}_{in} &\triangleq \{(i, j') : i \in V_1^a, j' \in V_2^a, 1 \le a \le m\} \\ \mathcal{M}_{across} &\triangleq \{(i, j') : i \in V_1^a, j' \in V_2^b, 1 \le a, b \le m, a \ne b\}, \end{aligned} \tag{4.2}$$

where $\mathcal{M}_{in}$ and $\mathcal{M}_{across}$ represent mappings within and across modules in two networks, respectively. We wish to show that eigenvector centrality scores assigned to $\mathcal{M}_{in}$ mappings in the EigenAlign algorithm are strictly larger than the ones assigned to $\mathcal{M}_{across}$ mappings. Suppose G_1 and G_2 have m blocks, each with size n. Thus, the total number of nodes in each network is mn. Let A represent the alignment network across G_1 and G_2, according to (2.3). This network has

m^2n^2 number of nodes, out of which mn^2 are within module mappings (i.e., $|\mathcal{M}_{in}| = mn^2$), and the rest are across module mappings (i.e., $|\mathcal{M}_{across}| = (m^2 - m)n^2$). Let $\bar{A}$ be the expected alignment matrix, where $\bar{A}(t_1, t_2) = \mathbb{E}[A(t_1, t_2)]$. To highlight key optimality conditions, in this section, we focus on the eigenvector analysis of the matrix $\bar{A}$. Similarly to Section 3.3, these properties can be extended to the alignment network A by using perturbation analysis of random matrices.

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 $0 < \epsilon \ll 1$. To simplify notation, we assume ϵ is sufficiently small so that its effect is negligible. Let v correspond to the leading eigenvector of the expected alignment matrix $\bar{A}$.

Theorem 3 *For the noiseless case ($p_e = 0$), if $p > q$, $\alpha > 1/q - 1$, and $m > 2$, then,*

$$v(i) > v(j), \quad \forall i \in \mathcal{M}_{in}, \forall j \in \mathcal{M}_{across}. \tag{4.3}$$

Proof A proof is presented in Section 8.3. ∎

The condition $p > q$ implies that the connectivity density of nodes in modules are larger than the one across modules. This condition is also essential in the spectral partitioning step of the EigenAlign+SDP algorithm to infer modules reliably. The condition $m > 2$ is a technical condition required in the proof of the strict inequality of (4.3). In Section 5.2, through simulations, we show that, this condition is not essential for the algorithm. The condition $\alpha > 1/q - 1$ guarantees that the expected alignment score of two mapping pairs where one belongs to $\mathcal{M}_{in}$ and the other one belongs to $\mathcal{M}_{across}$ is strictly larger than the one where both mappings belong to $\mathcal{M}_{across}$. More details on these optimality conditions can be found in Section 8.3.

Now, we consider the case when G_2 is related to a noisy version of G_1:

Theorem 4 *Suppose $0 \le p_e^2 \ll 1$ and $0 \le p_{e_2}^2 \ll 1$. Let $p > q$, $m > 2$ and $\alpha \gg 1/q$.*

- *If $q \le \frac{1}{4}$ and $p < \frac{1}{1+p_e}$,*
- *or, if $q > \frac{1}{4}$ and $p < min(\frac{1}{1+p_e}, \frac{6q-\sqrt{4q(4q-1)(1-q)}}{2(1+2q)})$,*

then,

$$v(i) > v(j), \quad \forall i \in \mathcal{M}_{in}, \forall j \in \mathcal{M}_{across}. \tag{4.4}$$

Proof A proof is presented in Section 8.4. ∎

If $p_e = 0$, the condition $p < \frac{1}{1+p_e}$ is simplified to $p < 1$. For higher noise level, this condition guarantees that correct within module mappings are assigned to higher scores than incorrect within-module ones. The additional constraint $p < \frac{6q-\sqrt{4q(4q-1)(1-q)}}{2(1+2q)}$ is required to guarantee that, the expected alignment score of two mapping pairs, one in $\mathcal{M}_{in}$ and the other one in $\mathcal{M}_{across}$, is strictly larger than the one where both mappings belong to $\mathcal{M}_{across}$. We illustrate Theorems 3 and 4 in Figure 4. More details on these sufficient optimality conditions can be found in Section 8.4.

Theorems 3 and 4 indicate that, eigenvector scores of within module mappings are strictly larger than the ones of across module mappings, in the expected alignment network. However, definitions of $\mathcal{M}_{in}$ and $\mathcal{M}_{across}$ are based on the permutation matrix P which relates networks G_1 and G_2 according to (3.11). In general, P may not be necessarily the optimal solution of the network alignment optimization (2.5). In the following, we provide conditions where the permutation matrix P in fact maximizes the expected objective function of the network alignment optimization (2.5).

Theorem 5 *Under the conditions of Theorem 3, if*

$$p \leq \frac{1+\sqrt{1+(\alpha^2-1)q}}{1+\alpha}, \tag{4.5}$$

then, the permutation matrix P maximizes the expected objective function of the network alignment optimization (2.5).

Proof A proof is presented in Section 8.5. ■

For instance, setting $\alpha = 1/q$ and $q \ll 1$, the condition of Theorem 5 results to the condition $p^2 \leq q$.

Theorem 6 *Under the conditions of Theorem 4, if*

$$p^2 \leq \frac{q(1-p_e)}{1+p_e^2}, \tag{4.6}$$

then, the permutation matrix P maximizes the expected objective function of the network alignment optimization (2.5).

Proof A proof is presented in Section 8.6. ■

Conditions of Theorems 5 and 6 are based on an upper bound on the parameter p. To explain this condition intuitively, suppose the permutation matrix P maps modules V_1^a in network G_1 to modules V_2^a in network G_2, where $1 \leq a \leq m$. In the case of large number of modules, the expected alignment score of the permutation matrix P is dominated by mapping-pairs (V_1^a, V_2^a) and (V_1^b, V_2^b), where $a \neq b$. These scores depend on the connectivity density across modules, namely q. On the other hand, consider an incorrect permutation matrix $\tilde{P}$ where $\frac{1}{nm}\|P-\tilde{P}\| > 0$. The alignment score of $\tilde{P}$ is upper bounded by scores of incorrect within module mappings which depend on p. To have a sufficient optimality condition, we wish the expected alignment score of P to be larger than the upper bound of the expected alignment score of $\tilde{P}$. More details on these optimality conditions are presented in Sections 8.5 and 8.6.

Remark 10 In the EigenAlign+SDP algorithm 2, errors can also happen in other steps of the algorithm including the network partitioning and semidefinite relaxation steps. In Section 5.2, we empirically analyze the performance of the proposed algorithm. Our results suggest that, the proposed method significantly outperforms existent network alignment techniques in aligning modular networks, while it has significantly lower computational complexity compared to standard convex-relaxation methods.

5 Performance Evaluation Over Synthetic Networks

We now compare the performance of the EigenAlign algorithm against other network alignment methods including IsoRank [3], NetAlign [20], Klau linearization [15] as well as an SDP-based method [17] through simulations. IsoRank is a global network alignment method which uses an iterative approach to align nodes across two networks based on their neighborhood similarities. NetAlign formulates the alignment problem in a quadratic optimization framework and uses message passing to approximately solve it. Klau linearization uses Lagrange multipliers to relax the underlying quadratic assignment problem. The SDP-based method [17] uses a convex relaxation of the underlying QAP based on matrix splitting. In our simulations, we use default parameters of these methods.

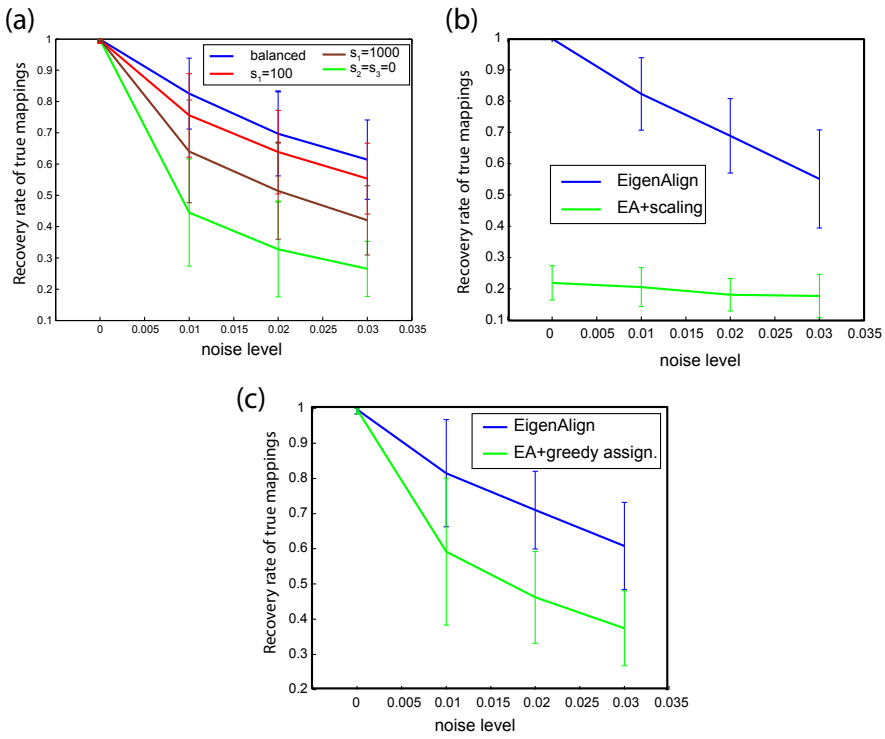

Figure 5: EigenAlign (EA) performance over power law graphs (a) with different alignment scores, (b) with alignment network scaling, and (c) with the greedy assignment method. At each point, simulations have been repeated 20 times.

5.1 Erdös-Rényi and Power-Law graphs

Here, we test alignment methods over two types of input networks:

- *Erdös-Rényi graphs [21]:* In this case, G_1 is a symmetric random graph where, $Pr[(i,j) = 1] = p$ (see an example in Figure 6-a).
- *Power law graphs [49]:* We construct G_1 as follows; we start with a random subgraph with 5 nodes. At each iteration, a node is added to the network connecting to θ existent nodes with probabilities proportional to their degrees. This process is repeated till the number of nodes in the network is equal to n (see an example in Figure 6-b).

G_2 is then formed according to (3.11). In our experiments, we consider two noise models (3.8) and (3.9). In the case of the power law network model, we use the density of G_1 as parameter p in the noise model II of (3.9). We denote p_e as the *noise level* in both models.

In (3.11), P can be an arbitrary permutation matrix. In our simulations, we choose P such that, $P(i, n-i+1) = 1$, for all $1 \le i \le n$, where n is the number of nodes in the network. Let $\tilde{P}$ represent the solution of a network alignment method in aligning networks G_1 and G_2. The recovery rate of true mappings is defined as,

$$1 - \frac{1}{2n}\|P - \tilde{P}\|. \tag{5.1}$$

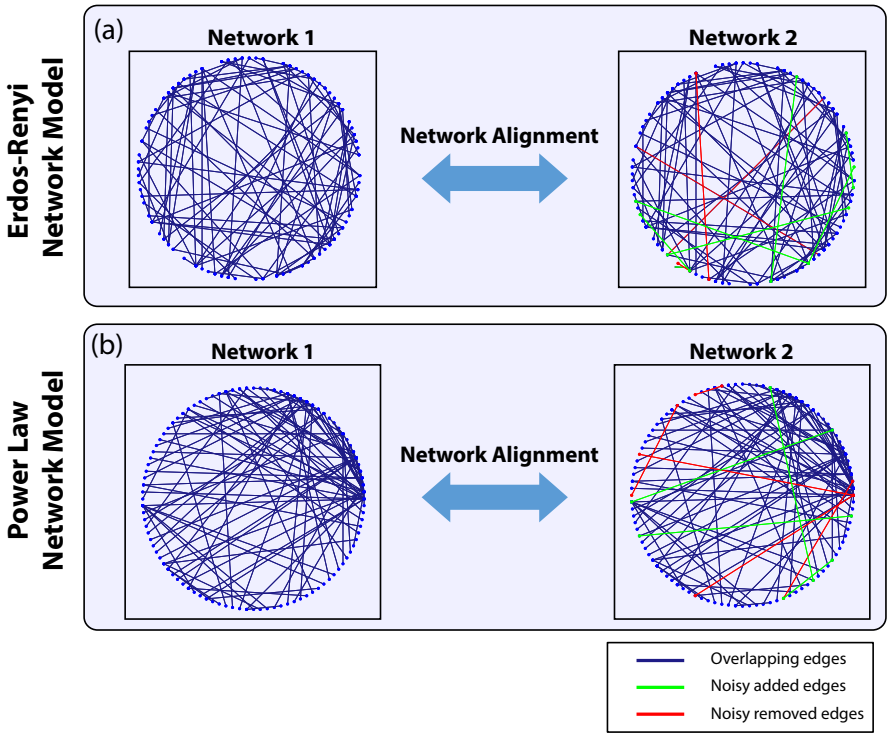

Figure 6: Examples of (a) Erdös-Rényi, and (b) power law networks used in Section 5.1.

As we explain in Proposition 2, this recovery rate is directly related to the number of inferred matches and mismatches in the network alignment optimization (2.5). Thus, we illustrate performance evaluations using this metric, noting that using other metrics such as the number of inferred matches/mismatches results in similar performance trends and conclusions. Moreover, in our simulations, all mappings across networks are considered to be possible (i.e., an unrestricted network alignment case).

Figure 5 illustrates individual contributions of different components of the EigenAlign algorithm by isolating multiple aspects of the algorithm. In these experiments, we use the power-law network structure with 50 nodes, $\theta = 3$, and noise model II. Performance trends and conclusions are similar for other cases as well. In EigenAlign Algorithm 1, we use $s_1 = \alpha + \epsilon$, $s_2 = 1 + \epsilon$, and $s_3 = \epsilon$, where $\epsilon = 0.001$ and,

$$\alpha = 1 + \frac{\#\text{ of mismatches}}{\#\text{ of matches}}. \tag{5.2}$$

This choice of α satisfies the required condition $\alpha > 1$. Moreover, by this selection of α, we have,

$$|s_1 - 1 - \epsilon|(\#\text{ of matches}) = |s_3 - 1 - \epsilon|(\#\text{ of mismatches}), \tag{5.3}$$

which makes a balance between matched and mismatched interaction scores across networks and leads to an improved performance of the method (Figure 5-a). Note that, ignoring mismatches

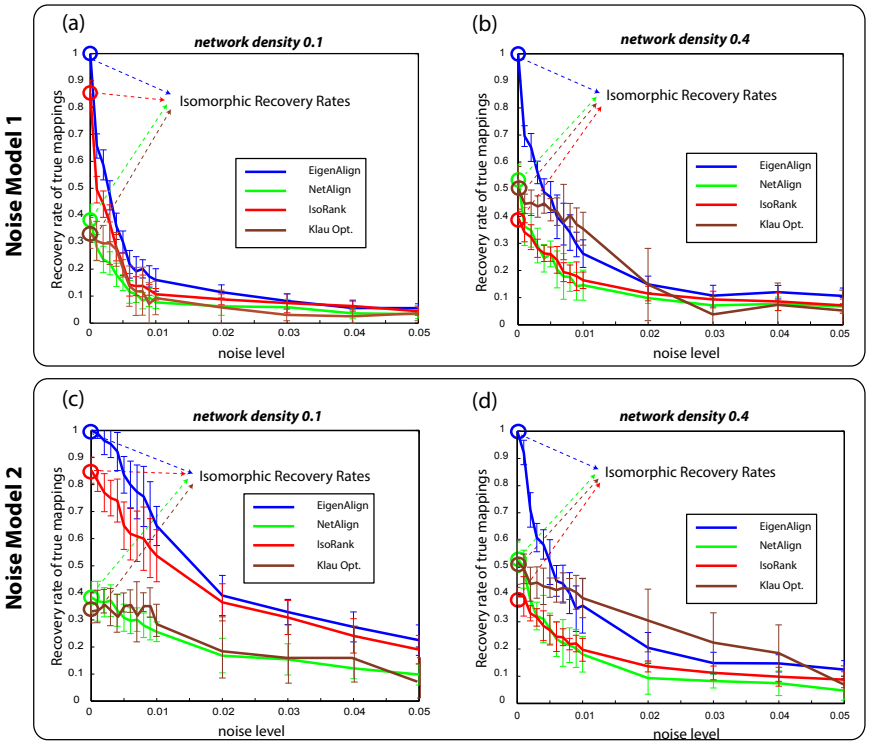

Figure 7: Performance evaluation of alignment methods over Erdös-Rényi graphs. EigenAlign outperforms IsoRank, NetAlign and Klau optimization in low and high network densities, in all considered noise levels, in both noise models. At each point, simulations have been repeated 10 times.

($s_2 = s_3 = 0$) leads to a significant decrease in the EigenAlign performance. Choosing other values of α can provide a way to adjust relative importance of matching and non-matching interactions in different applications. In general, this parameter can be tuned in different applications using standard machine learning techniques such as cross validations [50]. Figure 5-b illustrates the performance of the EigenAlign algorithm if we scale the alignment network by its node degrees (we replace $A_{i,j}$ by $A_{i,j}/d_i d_j$, where d_i is the degree of node i). As illustrated in this figure, this scaling destroys our global match-mismatch score assignments and leads to a significant decrease in the performance. Finally, Figure 5-c illustrates the performance of the EigenAlign algorithm if we replace the linear optimization of maximum weight bipartite matching with the greedy approach of Remark 7. As illustrated in this figure, this variation of the EigenAlign method decreases the performance, although it also decreases the computational complexity.

Next, we compare the performance of the EigenAlign method with the one of other network alignment methods over synthetic networks with $n = 100$ nodes. For large networks, the SDP-based method of [17] has high computational complexity, thus we excluded it from these experiments. Later, we will investigate the performance of the SDP-based method over small networks ($n \leq 50$).

Figure 7 shows the recovery rate of true mappings obtained by EigenAlign, IsoRank, NetAlign and Klau relaxation, over Erdös-Rényi graphs, and for both noise models. In the noiseless case (i.e., $p_e = 0$), the network alignment problem is simplified to the problem of graph isomorphism

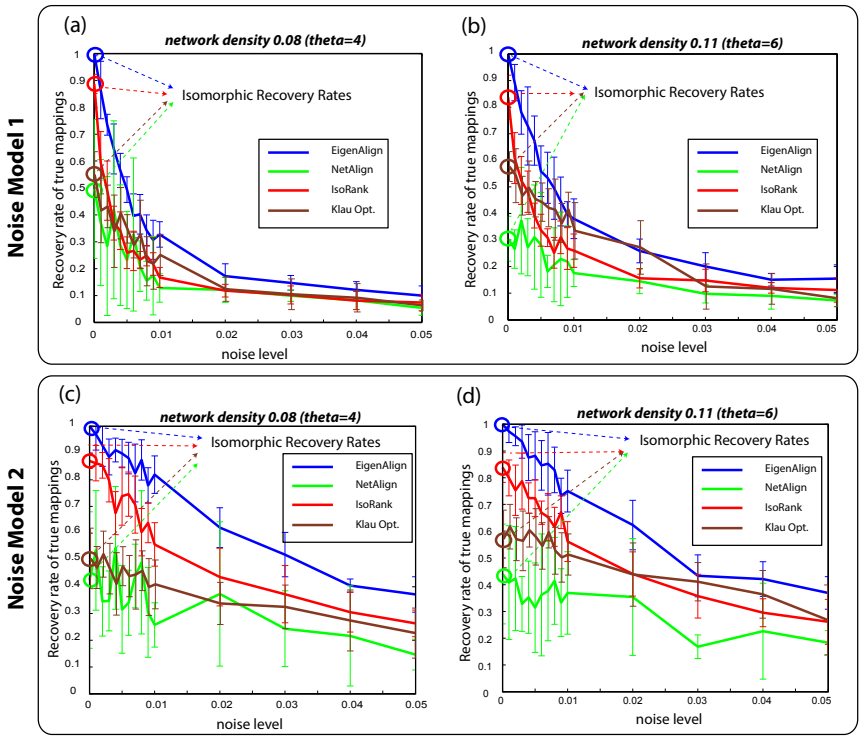

Figure 8: Performance evaluation of alignment methods over power law graphs. EigenAlign outperforms IsoRank, NetAlign, and Klau optimization in low and high network densities, in all considered noise levels, in both noise models. At each point, simulations have been repeated 10 times.

as explained in Section 3.3. In this case, the EigenAlign algorithm finds true mappings without error in both low and high network densities (i.e., the recovery rate of (5.1) is one.). EigenAlign continues to outperform other methods in all considered noise levels, in both low and high network densities.

Figure 8 shows the recovery rate of true mappings obtained by three considered methods over power law graphs, and for both noise models. Similarly to the case of Erdös-Rényi graphs, EigenAlign outperforms other methods in all considered noise levels and network densities. Notably, in noiseless cases, EigenAlign finds true isomorphic mappings across networks without error.

Figure 9 depicts the average running time of these methods over an Erdös-Rényi graph with density $p = 0.4$, and under the noise model I. All methods have been run on the same machine and each experiment has been repeated 10 times. In these experiments, EigenAlign has the second best runtime, after IsoRank, outperforming NetAlign and Klau relaxation methods. Notably, the running time of Klau optimization method increases drastically as network size grows.

Next, we consider small networks with $n = 50$ nodes so that we are able to include a computationally expensive SDP-based method of [17] in our experiments. Note that, the method proposed in [17] only provides a bound on the value of the underlying QAP objective function and it does not provide a feasible solution. In order to obtain a feasible solution for the network alignment optimization, we use the SDP-based solution in a maximum weight bipartite matching optimization.

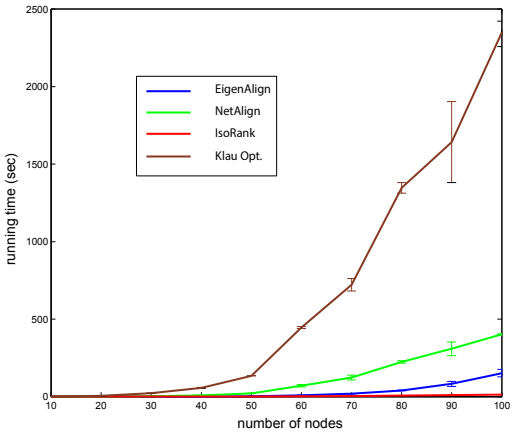

Figure 9: Running time of network alignment methods over Erdös-Rényi graphs.

Figures 12-a,b illustrate the recovery rate of true mappings obtained by five considered methods over both Erdös-Rényi and power law graphs, under the noise model II. Performance trends under the noise model I is similar. As illustrated in these figures, the computationally expensive SDP-based method outperforms other network alignment methods significantly, while it has the highest running time (Figure 13). Notably, in these experiments, EigenAlign outperforms other methods except the SDP-based one consistent with the cases illustrated in Figures 7 and 8. These results inspired us to propose the EigenAlign+SDP method to both have high recovery rate and low computational complexity (see Section 4). In the next section, we focus on the performance evaluation of the EigenAlin+SDP method over modular network structures.

5.2 Modular Network Structures

We then asses the performance of the proposed EigenAlign+SDP method in Section 4 in aligning networks with modular structures. To be able to compare the performance of the proposed method with the one of the SDP-based method [17], we only consider small modular networks with 50 nodes, having two equal-size modules. As discussed in Section 5.1, the SDP-based method of [17] has high computationally complexity and using it for large networks is not practically feasible. The key idea of the EigenAlign+SDP method is to use the EigenAlign solution to split the large QAP into smaller sub-problems, enabling the use of the SDP-based relaxation method over each sub-problem. Moreover, the SDP-based method can be run in parallel over each sub-problem.

Here, we consider network and noise models described in Section 4. In our evaluations, we consider EigenAlign, SDP, and EigenAlign+SDP methods. Moreover, we use the performance evaluation metric introduced in Section 5.1. Figures 12-a,b illustrates the recovery rate of true mappings of different methods, in various noise levels, and for different set of parameters p and q (the density of edges within and across modules, respectively). The average running time of the methods is depicted in Figure 13. As it is illustrated in these figures, while the recovery rate of the EigenAlign+SDP method is close to the one of the SDP-based one, it has significantly lower computational complexity, enabling its use for large complex networks.

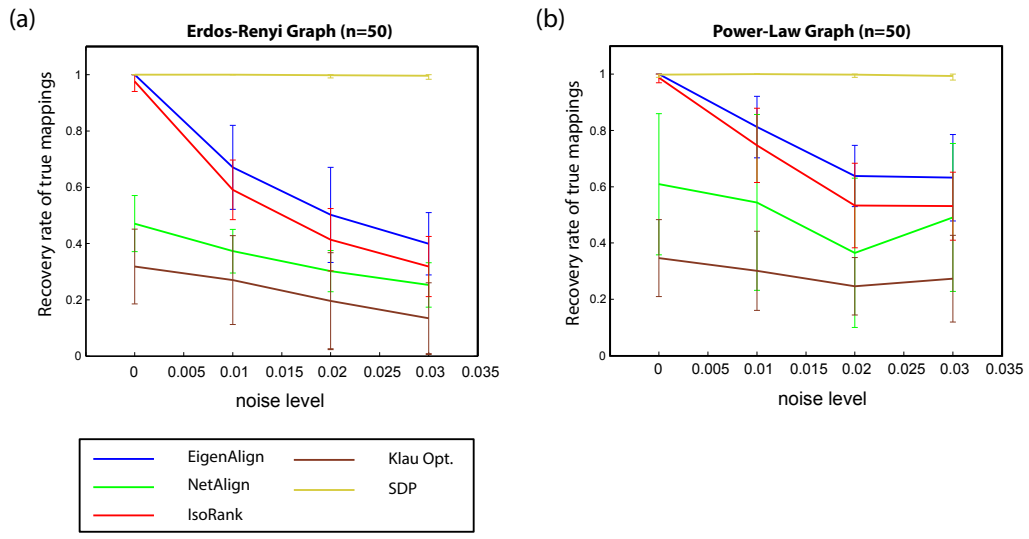

Figure 10: Performance evaluation of network alignment methods over (a) Erdös-Rényi ($p = 0.2$), and power law ($\theta = 3$) graphs, with $n = 50$. The computationally expensive SDP-based method outperforms other methods, while EigenAlign has the second best performance. At each point, simulations have been repeated 20 times.

6 Inference of Regulatory Networks in Human, Fly and Worm

Having illustrated the efficiency of the proposed network alignment algorithm, both theoretically and through simulations, we wish to use EigenAlign and other network alignment methods to compare the structure of regulatory networks across different species. However, the paucity of comprehensive catalogs of regulatory genomics datasets has hindered these studies in animal genomes. 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 the next section, we will compare the structure of these inferred networks using EigenAlign and other 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 [51]. 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 [52–54]. 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 [55, 56]. This is especially important because animal genomes, as fly, worm, and mouse are routinely used as models for human disease [57, 58].

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 [59] and gene expression levels [60, 61] 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

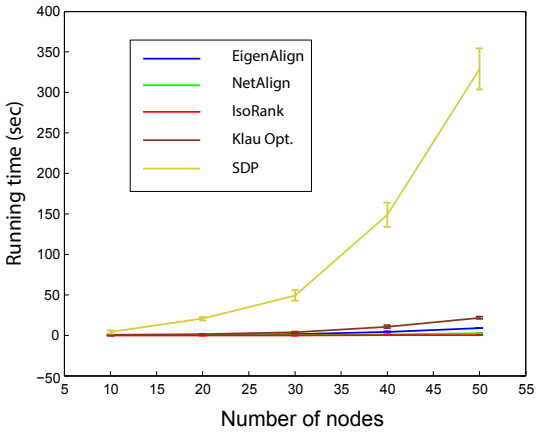

Figure 11: Running time of network alignment methods over Erdös-Rényi graphs with $n = 50$ and $p = 0.2$. The SDP-based method has significantly higher computational complexity compared to other network alignment techniques, which prohibits its use over large graphs. At each point, simulations have been repeated 20 times.

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 [54, 62]. 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 14-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 [54] 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 [62, 63]. However, integration of these three networks [53, 64–66] 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 two integration approaches: one approach combines interaction ranks across networks, while the other is based on mapping edge weights to interaction probabilities and then combining interaction likelihoods across input networks. Inferred regulatory interactions using both rank-based and likelihood-based integration methods show significant overlap with known interactions in human and fly, indicating

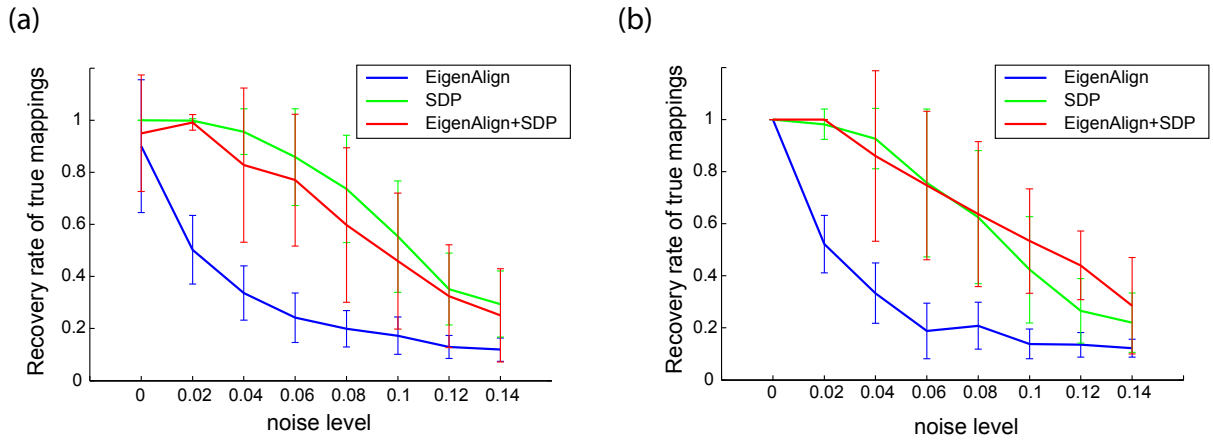

Figure 12: Performance evaluation of network alignment methods over modular network structures with $n = 20$, $m = 2$, $p = 0.2$, and (a) $q = 0.05$, (b) $q = 0.01$. The recovery rate of the EigenAlign+SDP method is close to the one of the SDP-based method, while it has significantly lower computational complexity.

the accuracy and robustness of the used inference pipeline. In the following, we explain our network inference framework with more details.

6.1 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 [51, 67–69]. There are several methods to infer regulatory networks using gene expression profiles [66]. 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 14-a): one method is CLR [60] (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 [61] (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. GENEI3 has been recognized as the top-performing expression based inference method in the DREAM5 challenge [66].

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 [70], 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,

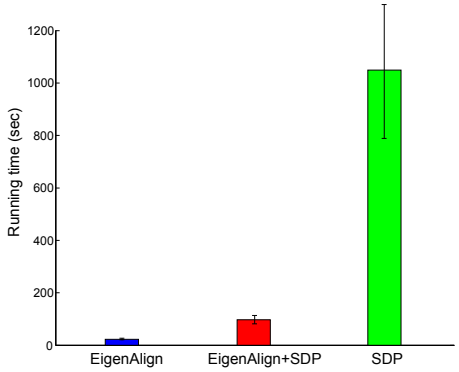

Figure 13: Average running time of network alignment methods over modular network structures with $n = 20$, $m = 2$, $p = 0.2$, and $q = 0.05$. The average running time of the SDP+EigenAlign algorithm is significantly lower than the one of the SDP-based method. Simulations have been repeated 20 times.

the quality of inferred expression networks are lower than the one for physical networks [62]. This can be partially owing to indirect effects and transitive interactions in expression-based regulatory networks [66].

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 [59], while in the second approach, we use experimentally defined TF binding occupancy profiles from ChIP assays of ENCODE and modENCODE [54,62]. 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 [59], 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 14-b). We use a fixed window of 5kb around the transcription start site (TSS) in human and 1kb 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.

6.2 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 [62], 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

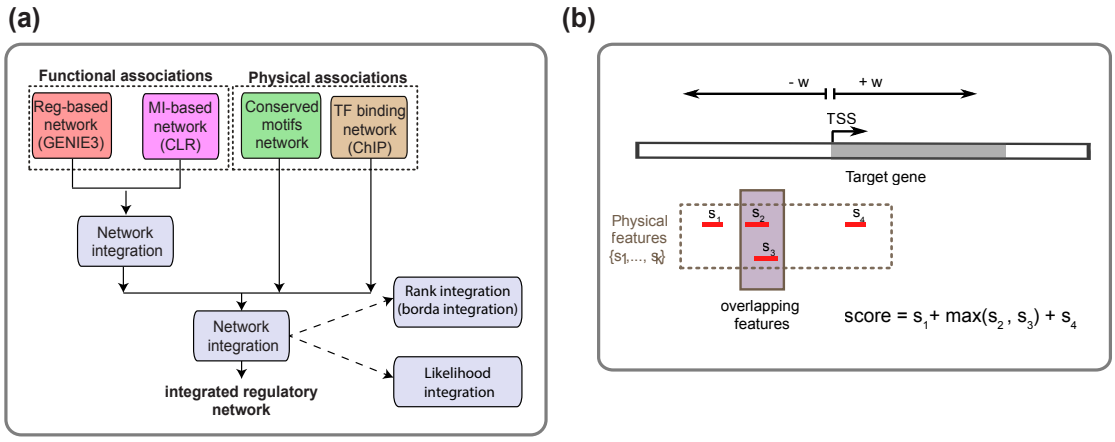

Figure 14: (a) The proposed framework to infer integrative regulatory networks. (b) The proposed framework to infer physical feature-specific regulatory networks.

regulatory networks compared to physical ones, they include many false-positive edges partially owing to indirect information flows [63]. To overcome these limitations, we therefore integrate these feature-specific regulatory interactions into a single integrated network [53, 64–66] (Figure 14-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 $w^l_{i,j}$ 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 the following techniques to infer integrated networks:

Rank-based (borda) integration: In this approach, integrative weights are computed as follows:

$$w_{i,j} = 1/K \sum_{l=1}^{K} r^l_{i,j}, \tag{6.1}$$

where $r^l_{i,j}$ 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 $w^l_{i,j} = 0$, then $r^l_{i,j} = 0$) [66]. Moreover, ties are broken randomly among edges with same weights.

Likelihood-based integration: In this approach, integrative weights are computed as follows,

$$w_{i,j} = 1/K \sum_{l=1}^{K} -\log\left(1 - \varrho p^l_{i,j}\right), \tag{6.2}$$

where $p^l_{i,j}$ represents the probability that edge $i \rightarrow j$ exists in the input network l, defined as $p^l_{i,j} \triangleq Pr(w < w^l_{i,j})$. ϱ is a number close to one (e.g., $\varrho = 0.99$) to have a well-defined $\log(.)$ function when the edge probability is one. This approach can be considered as a maximum likelihood estimator of integrative weights if input networks and their interactions are assumed to be independent, and empirical distributions of input edge weights are sufficiently close to actual distributions.

We find that, top-ranking integrative interactions in human and fly networks are primarily supported by ChIP and motif evidences specially in the rank-based integrative networks, while worm

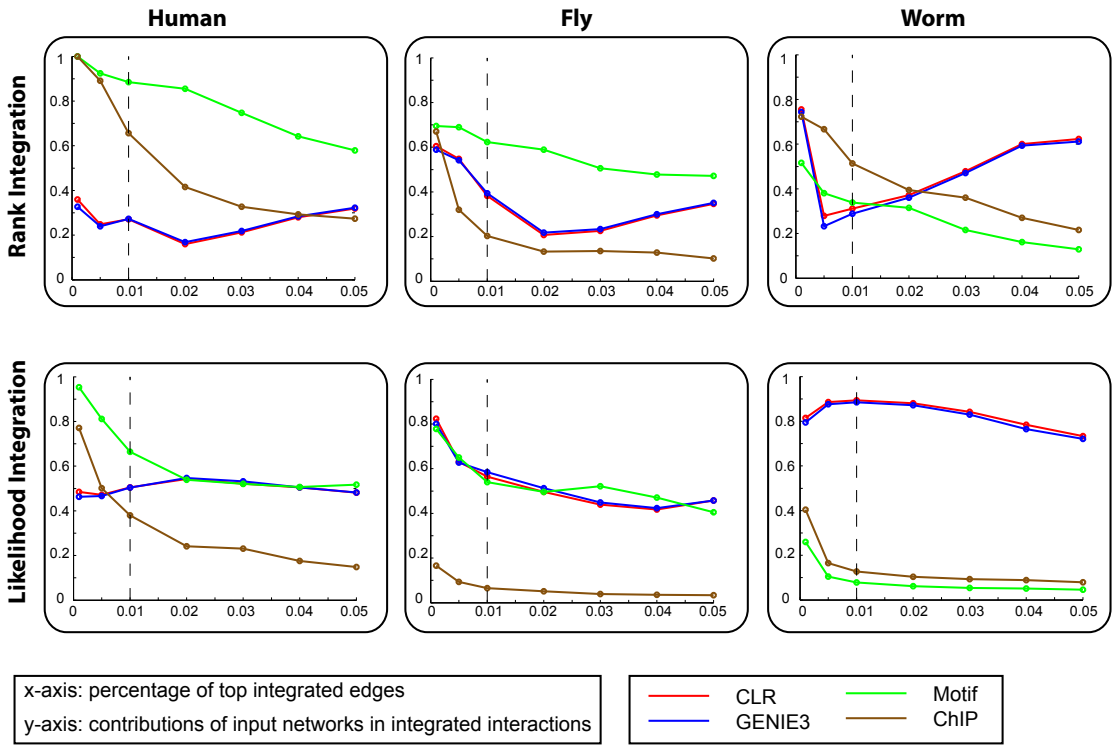

Figure 15: Contributions of input feature-specific networks in integrated interactions.

interactions are primarily supported by co-expression edges, consistent with the lower coverage of worm ChIP and motif interactions (Figure 15).

To validate inferred integrated networks, we use known interactions in TRANSFAC [26], REDfly [27] and EdgeDB [28] 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 16). 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 16 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, both integrative networks (rank-based and likelihood-based) outperform feature-specific networks in all three species. In fact, AUROC values of rank-based and likelihood based integrative networks are 0.58 in human, 0.62 and 0.61 in fly, and 0.52 and 0.51 in worm, respectively. According to the AUPR metric and using the same cut-off, the likelihood-based integrative network outperforms other networks in human

	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.

and fly species. AUPR values of rank-based and likelihood based integrative networks are 0.019 and 0.017 in human, 0.047 and 0.045 in fly, and 0.037 and 0.035 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.

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 the rank-based 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. In the likelihood-based integrative networks, the median number of targets for each TF are 478, 400 and 861, with a median of 136, 29 and 41 regulators per gene in in human, fly and worm, respectively.

7 Comparative Analysis of Gene Regulatory Networks across Human, Fly and Worm

In this section, we apply network alignment methods to compare gene regulatory networks of human, fly, and worm, three of the most distant and deeply studied metazoan species (Figure 17-a). We use regulatory networks which we inferred in Section 6 by integrating genome-wide functional and physical genomics datasets from ENCODE and modENCODE consortia. In this section, we focus on analyzing ranked-based integrated regulatory networks, while all arguments can be extended to likelihood-based networks as well.

We use homolog mappings across genes of human, fly and worm provided by the ENCODE and modENCODE consortia [71]. Note that, human homologs refer to homologous genes in fly and worm, while fly/worm homologs are defined solely in relation to human. Homolog mappings across species are based on gene sequence similarities over corresponding gene trees [71]. However, owing to multiple duplications and losses, homolog mappings are not bijective necessarily. For example, one gene in human can be homologous to multiple genes in fly and vice versa (see an example in Figure 17-b). Comparative network analysis in evolutionary studies often requires having a one-to-one mapping across genes of two networks. In this section, we use network alignment methods

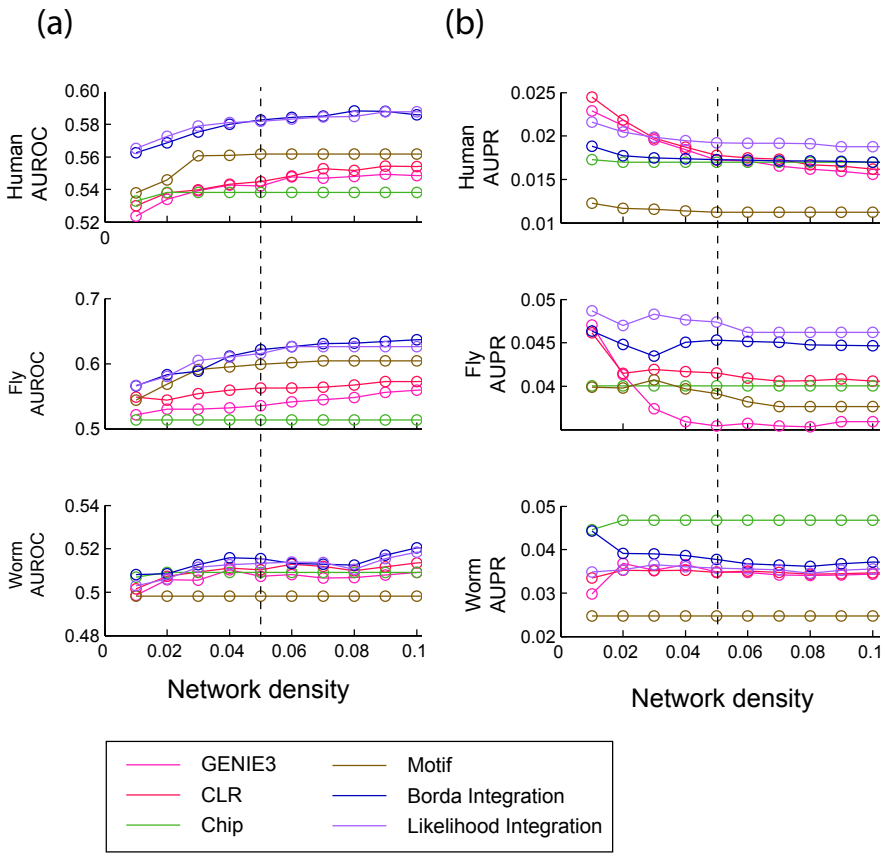

Figure 16: AUROC and AUPR scores of feature-specific and integrated regulatory networks in human, fly and worm species.

to infer bijective mappings as a subset of homolog mappings across species (Figure 17-b). An optimal bijective mapping across networks of different species causes the most number of matched interactions (i.e., interactions that exists in both networks) and the least number of mismatched interactions (i.e., interactions that only exist in one of the networks).

We assess the performance of three network alignment methods (EigenAlign, NetAlign, IsoRank) in comparative analysis across gene regulatory networks of human, fly and worm. We excluded the network alignment method based on Klau optimization [15] from our analysis in this section owing to its high computational complexity, and its low performance over synthetic networks of Section 5. Moreover, as we discussed in Section 5, the SDP-based method of [17] has high-computational complexity, which prohibits its use in aligning large regulatory networks. We use EigenAlign, IsoRank and NetAlign methods with the setup and parameters similarly to Section 5.

Unlike EigenAlign, IsoRank and NetAlign 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-regulation networks using inferred regulatory networks of Section 6, by connecting genes when their parent TFs have an overlap larger than 25%. This results in undirected binary co-regulation networks in human, fly, and worm, with 19,221,

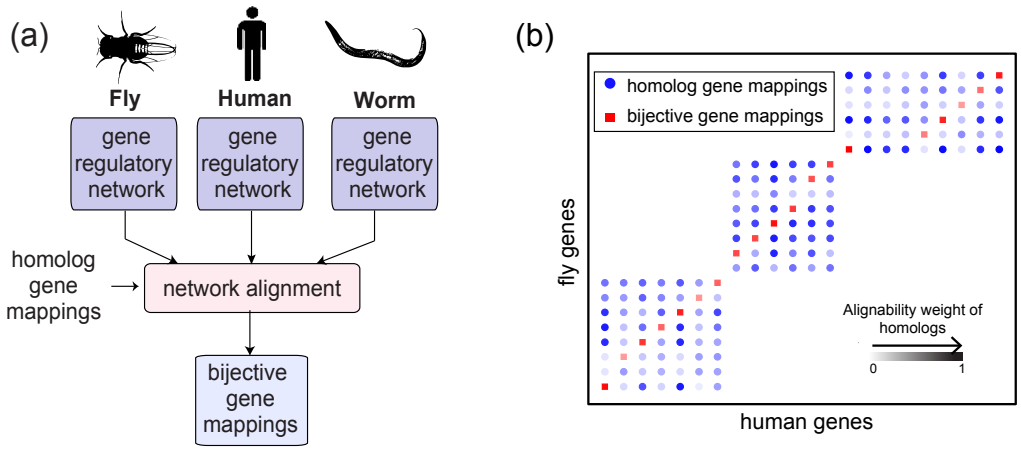

Figure 17: (a) A comparative analysis framework across human, fly and worm regulatory networks. (b) An application example of a network alignment method (EigenAlign) over three gene families across human-fly.

13,642, and 19,296 nodes, and 13.9%, 3.5%, and 4.2% edge densities, respectively.

By application of network alignment methods, we infer bijective mappings across human-fly and human-worm networks. There are numerous mismatched interactions across networks of distal species partially owing to extensive gene functional divergence due to processes such as gene duplication and loss. In this situation, it is critical to consider both matches and mismatches in the network alignment optimization. Unlike existent network alignment algorithms which completely ignore mismatches across networks, EigenAlign considers both matches and mismatches across networks in a balanced way by setting alignment scores according to (5.2). Figures 18-a,b illustrate the number of matched (conserved) and mismatched (error) interactions across human-fly and human-worm networks, inferred using different methods. As it is illustrated in these figures, in both human-fly and human-worm comparisons, EigenAlign significantly outperforms other methods in terms of causing fewer number of mismatches (errors) across networks, while its performance is close to the best one in terms of the number of matches (overlaps). Figure 18-c illustrates the match-mismatch ratio for different network alignment methods, in both human-fly and human-worm comparisons. As it is illustrated in this figure, EigenAlign outperforms other methods significantly indicating that, it finds mappings across networks which maximize the number of matches and minimize the number of mismatches simultaneously. In all cases, the number of conserved interactions is statistically significant ($p - value < 0.01$), indicating gene evolution has occurred in a way to preserve regulatory pathways across these species. p-values are computed using a rank test by randomly selecting bijective mappings from gene homolog mappings while keeping network structures the same.

Next, we assess conservation of gene centralities across different networks. Centrality measures over regulatory networks provide useful insights on functional roles of genes in regulatory processes [72–74]. Therefore, comparative analysis of gene centrality measures across species can shed light on how evolution has preserved or changed regulatory patterns and functions across distal species. A suitable alignment should map central nodes in one network to central nodes in the other network (Figure 19-a). Here, we consider three centrality measures: degree centrality,

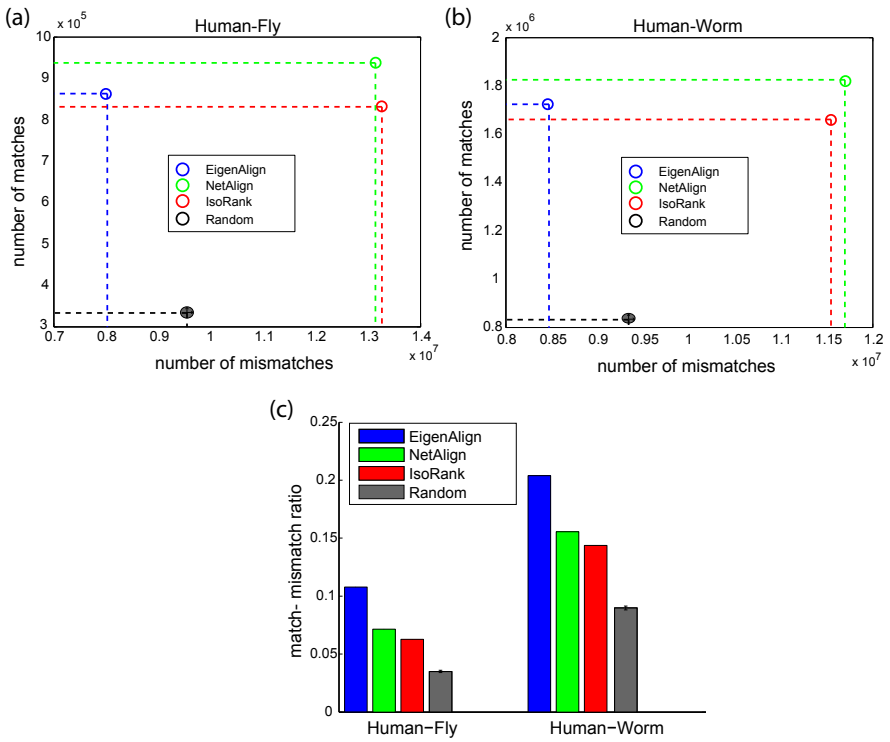

Figure 18: The number of matched and mismatched interactions inferred using different network alignment methods across (a) human-fly and (b) human-worm networks. (c) An average match-mismatch ratio for different network alignment methods, in both human-fly and human-worm comparisons.

eigenvector centrality, and the page-rank centrality with the damping factor 0.85. In Figure 19, we illustrate centrality correlation of aligned genes using different network alignment methods, across both human-fly and human-worm networks. As it is illustrated in this figure, the bijective mapping inferred by EigenAlign preserves different node centralities across networks significantly better than other tested methods. That is, EigenAlign is more likely to map central nodes in one network to central nodes in the other network. Finally, we note that, the centrality conservation across human-fly networks is significantly larger than the one across human-worm networks, partially owing to a larger evolutionary distance between human and worm compared to the one between human and fly.

Next, we examine enrichment of different biological processes over conserved subgraphs inferred by different alignment methods across human-fly and human-worm. To do this, we use genome ontology (GO) processes that are experimentally verified and have sizes in the range of 50 and 500 genes. For each GO category, we compute the number of matched and mismatched interactions using bijective mappings of different network alignment methods across human-fly and human-worm networks. Figure 20-a illustrates the average match-mismatch ratio over all considered GO categories, across both human-fly and human-worm networks. As it is illustrated in this figure, the EigenAlign solution preserves the connectivity pattern of genes in each GO category across these species better than other network alignment methods. This is consistent with the overall match-mismatch ratio gain of the EigenAlign method across human-fly and human-worm networks,

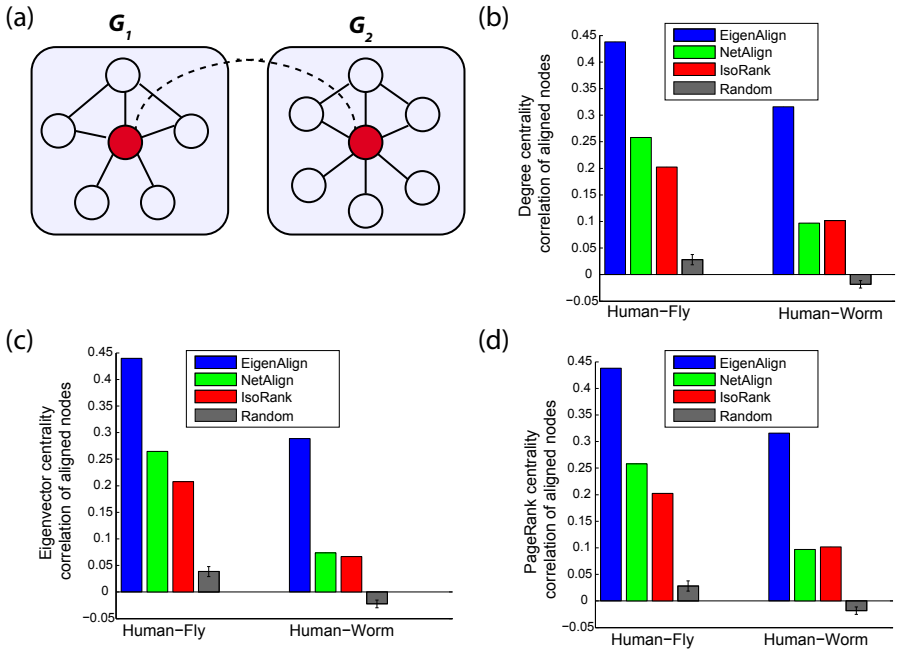

Figure 19: (a) A suitable alignment should map central nodes in one network to central nodes in the other network. Correlation of (b) degree centrality, (c) eigenvector centrality, and (d) page-rank centrality measures of aligned genes using different network alignment methods, in both human-fly and human-worm comparisons.

demonstrated in Figure 18-c. Note that, the average match-mismatch ratios across GO processes (Figure 20-a) are significantly higher than the ones across entire networks (Figure 18-c). This may indicate that, connectivity patterns among genes involved in similar processes are better preserved compared to the rest of the regulatory network. However, note that, this enrichment of GO processes across species may be partially owing to biases in the study of these genes in the literature [75]. Thus, additional experimental validations are required which is beyond the scope of this paper.

Next, we focus on GO categories whose regulatory patterns are (partially) conserved across these species, according to mappings of *at least* one of the considered network alignment methods. To do this, we select GO categories whose match-mismatch ratio is larger than or equal to one. Considering the small match-mismatch ratio of entire regulatory networks (Figure 18-c), this measure selects processes that have significantly higher conservation than the one of randomly selected gene sets. We also limit processes to the ones with at least 10 conserved interactions to eliminate processes whose conservation may not be statistically significant. Figure 20-b illustrates the average match-mismatch ratio over partially conserved GO processes, across both human-fly and human-worm networks. As illustrated in this figure, most of the (partially) conserved GO processes are identified by EigenAlign mappings across networks. In fact, EigenAlign identifies 6 processes as partially conserved ones across human-fly, while this number for NetAlign and IsoRank are 3 and 2, respectively. The conserved processes of EigenAlign solution include an immune system process, embryo development, actin filament organization, and cellular response to endogenous stimulus. Across human-worm networks, EigenAlign identifies 5 partially conserved processes, while this number for NetAlign and IsoRank are 1, and 0, respectively. In this case, conserved processes of

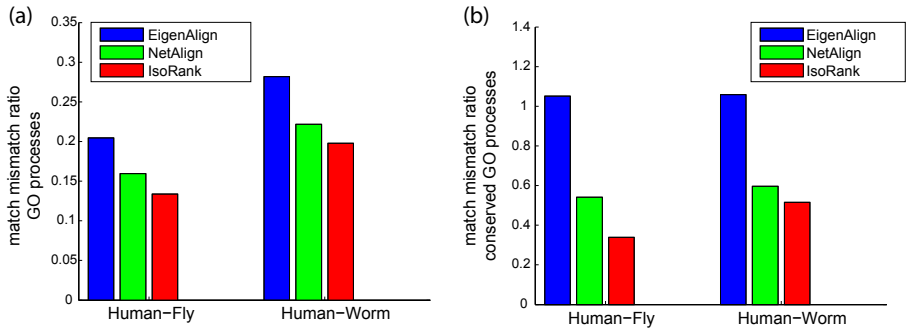

Figure 20: An average match-mismatch ratio across both human-fly and human-worm networks over (a) all , and (b) partially conserved, considered GO categories.

the EigenAlign solution include different compound catabolic processes. Notably, by considering a less restrictive match-mismatch ratio of 0.8, all conserved processes of other network alignment methods are inferred using the EigenAlign method as well. Finally, we note that, the solution provided by the EigenAlign method can be used in designing future experiments to extend GO catalogs by using gene regulatory networks and cross-species information.

8 Proofs

In this section, we present proofs of Theorem 2 and Corollary 1. First, in Section 8.1, we present proof of Corollary 1 to highlight key ideas used in the proof. Then, in Section 8.2, we present the proof of Theorem 2.

8.1 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 $\{1 \leftrightarrow 1', 2 \leftrightarrow 2', \ldots, n \leftrightarrow n'\}$ (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 $1 \le i,j \le n$. Recall that Y is a vector of length kn which has weights for all possible mappings $(i,j') \in \mathcal{R}$. 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., $y(1) = y_{1,1'}, y(2) = y_{2,2'}, \ldots, y(n) = y_{n,n'}$.
- The remaining $(k-1)n$ indices of $\mathbf{y}$ correspond to incorrect mappings. e.g., $y(n+1) = y_{1,2'}, y(n+2) = y_{1,3'}, \ldots, y(kn) = y_{r,s'}$ $(r \neq s)$.

Therefore, we can write,

$$\mathbf{y} = \left[\frac{\mathbf{y}_1}{\mathbf{y}_2}\right],$$

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: $\mathcal{S}_1 = \{1, 2, \ldots, n\}$ and $\mathcal{S}_2 = \{n+1, n+2, \ldots, kn\}$. The alignment matrix A for graphs G_1 and G_2 can be characterized using equation (2.3) as follows:

$$A(t_1, t_2) = \begin{cases} (\alpha+1)G_1(i,j)G_2(i',j') - G_1(i,j) - G_2(i',j') + 1 + \epsilon, & \text{if } t_1 \sim (i,i'), t_2 \sim (j,j'), \\ & t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2. \\ (\alpha+1)G_1(i,j)G_2(r',s') - G_1(i,j) - G_2(r',s') + 1 + \epsilon, & \text{if } t_1 \sim (i,r'), t_2 \sim (j,s'), \\ & t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2. \\ 1 + \epsilon, & \text{if } t_1 = t_2, \end{cases} \tag{8.1}$$

where notation $t_1 \sim (i, r')$ means that, row (and column) index t_1 of the alignment matrix A corresponds to the mapping (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 (8.1) can be written as,

$$A(t_1, t_2) = \begin{cases} (\alpha+1)G_1(i,j)^2 - 2G_1(i,j) + 1 + \epsilon, & \text{if } t_1 \sim (i,i'), t_2 \sim (j,j'), \\ & t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2. \\ (\alpha+1)G_1(i,j)G_1(r,s) - G_1(i,j) - G_1(r,s) + 1 + \epsilon, & \text{if } t_1 \sim (i,r'), t_2 \sim (j,s'), \\ & t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2. \\ 1 + \epsilon, & \text{if } t_1 = t_2. \end{cases} \tag{8.2}$$

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 $A(t_1, t_2)$.

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

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

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

$$v_{1,1} = v_{1,2} = \ldots = v_{1,n} = v_1^*,$$
$$v_{2,1} = v_{2,2} = \ldots = v_{2,(k-1)n} = v_2^*,$$

Moreover, if $n \to \infty$*, then,*

$$\frac{v_1^*}{v_2^*} > \beta, \tag{8.3}$$

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

Proof Since $G_1(i,j)$ is a Bernoulli random variable which is one with probability p, equation (8.4) leads to:

$$\bar{A}(t_1, t_2) = \begin{cases} (\alpha-1)p + 1 + \epsilon, & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2, \\ (\alpha+1)p^2 - 2p + 1 + \epsilon, & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2, \\ 1 + \epsilon, & \text{if } t_1 = t_2. \end{cases} \tag{8.4}$$

Define $a \triangleq (\alpha-1)p+1+\epsilon$ and $b \triangleq (\alpha+1)p^2-2p+1+\epsilon$.

Since bv is an eigenvector of $\bar{A}$, we have,

$$\bar{A}\mathbf{v} = \lambda\mathbf{v}, \tag{8.5}$$

where λ is the corresponding eigenvalue of bv. Therefore,

$$\bar{A}\mathbf{v} = \begin{bmatrix} a\sum_i v_{1,i} + b\sum_j v_{2,j} + (1+\epsilon-a)v_{1,1} \\ \vdots \\ a\sum_i v_{1,i} + b\sum_j v_{2,j} + (1+\epsilon-a)v_{1,n} \\ \hline b\sum_i v_{1,i} + b\sum_j v_{2,j} + (1+\epsilon-a)v_{2,1} \\ \vdots \\ b\sum_i v_{1,i} + b\sum_j v_{2,j} + (1+\epsilon-a)v_{2,(k-1)n} \end{bmatrix} = \lambda \begin{bmatrix} v_{1,1} \\ \vdots \\ v_{1,n} \\ \hline v_{2,1} \\ \vdots \\ v_{2,(k-1)n} \end{bmatrix}. \tag{8.6}$$

Therefore,

$$\begin{aligned} a\sum_i v_{1,i} + b\sum_j v_{2,j} &= v_{1,r}(\lambda+a-1-\epsilon), \qquad \forall 1 \le r \le n, \\ b\sum_i v_{1,i} + b\sum_j v_{2,j} &= v_{2,s}(\lambda+b-1-\epsilon), \qquad \forall 1 \le s \le (k-1)n. \end{aligned} \tag{8.7}$$

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 (8.7) leads to,

$$\begin{aligned} v_{1,1} = v_{1,2} = \ldots = v_{1,n} &= v_1^*, \\ v_{2,1} = v_{2,2} = \ldots = v_{2,(k-1)n} &= v_2^*. \end{aligned} \tag{8.8}$$

Using equations (8.7) and (8.8), we have,

$$\begin{cases} anv_1^* + b(k-1)nv_2^* = v_1^*(\lambda+a-1-\epsilon) \\ bnv_1^* + b(k-1)nv_2^* = v_2^*(\lambda+b-1-\epsilon). \end{cases} \tag{8.9}$$

We choose ϵ so that $\lambda + b\big(1-(k-1)n\big) - 1 - \epsilon \neq 0$. We will show later in this section that any sufficiently small value of ϵ satisfies this inequality. Further, according to PerronFrobenius Theorem 1, $v_{1,i} > 0$ and $v_{2,j} > 0$, for all i and j. Under these conditions, solving equation (8.9) leads to:

$$(\lambda-\lambda_a)(\lambda-\lambda_b) = b^2(k-1)n^2, \tag{8.10}$$

where,

$$\begin{cases} \lambda_a = (n-1)a + 1 + \epsilon, \\ \lambda_b = \big((k-1)n-1\big)b + 1 + \epsilon. \end{cases} \tag{8.11}$$

Equation (8.10) has two solutions for λ. However, since λ is the largest eigenvalue of the expected alignment matrix $\bar{A}$, we choose the largest of the roots. 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 (8.10) and (8.11), 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 $n \to \infty$, equation (8.9) implies,

$$\frac{v_1^*}{v_2^*} = \frac{\lambda}{bn} - k + 1, \tag{8.12}$$

where λ is the largest root of equation (8.10). For sufficiently large n,

$$\frac{v_1^*}{v_2^*} = \frac{1}{2}[(\frac{a}{b} - k + 1) + \sqrt{(\frac{a}{b} - k + 1)^2 + 4k - 4}]. \tag{8.13}$$

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,

$$\frac{a}{b} > 1 + \Delta k > 1 + \Delta(1 + \frac{k-1}{1+\Delta}) = 1 + \Delta + \frac{\Delta}{\Delta + 1}(k-1). \tag{8.14}$$

Using inequality (8.14) in (8.13), we have,

$$\begin{aligned} \frac{v_1^*}{v_2^*} &> \frac{1}{2}\left[\frac{(1+\Delta)^2 - k + 1}{1+\Delta} + \frac{\sqrt{((1+\Delta)^2 - k + 1)^2 + 4(k-1)(1+\Delta)^2}}{1+\Delta}\right] \\ &= 1 + \Delta. \end{aligned} \tag{8.15}$$

This completes the proof of Lemma 3. ■

Remark 11 *In Lemma 3, if $kp = c_1$, where $c_1 \ll k$ and $k \gg 1$, choosing $\alpha = c_2k^2$ results in $\Delta \approx ck$ where $c = \frac{c_1c_2}{c_1^2c_2+1} > 1$.*

If $\mathbf{v}$ is an eigenvector with unit norm, we have:

$$\|\mathbf{v}\| = 1 \Rightarrow nv_1^* + (k-1)nv_2^* = 1. \tag{8.16}$$

By using this equation and Lemma 3, for sufficiently large n, we have:

$$\begin{aligned} v_1^* &= \frac{\beta}{\sqrt{\beta^2 + k}}\frac{1}{\sqrt{n}}, \\ v_1^* &= \frac{1}{\sqrt{\beta^2 + k}}\frac{1}{\sqrt{n}}. \end{aligned} \tag{8.17}$$

Since edges of network G_1 are random, the alignment network is a also random matrix. Let Ω be a set of outcomes of edge variables of network G_1, and let $A(\Omega)$ be the corresponding alignment network. Let $\mathbf{v}(\Omega)$ represent an eigenvector of $A(\Omega)$ with the largest eigenvalue.

Lemma 4 *Let* $\mathbf{v}(\Omega)$ *be a unite norm eigenvector with the largest eigenvalue of the alignment matrix* $A(\Omega)$*. Suppose*

$$\mathbf{v}(\Omega) = \begin{bmatrix} \mathbf{v}_1(\Omega) \\ \hline \mathbf{v}_2(\Omega) \end{bmatrix},$$

where $\mathbf{v}_1(\Omega)$ *and* $\mathbf{v}_2(\Omega)$ *are vectors of length* n *and* kn*, respectively. Let* $\bar{\mathbf{v}} = \mathbb{E}[\mathbf{v}(\Omega)]$*. Then,*

- *(i) For all* $1 \le i \le n$, $\mathbb{E}[v_{1,i}(\Omega)] = \bar{\mathbf{v}}_1$, *and* $\sigma^2_{v_{1,i}(\Omega)} = \sigma_1^2$.
- *(ii) For all* $1 \le j \le (k-1)n$, $\mathbb{E}[v_{2,j}(\Omega)] = \bar{\mathbf{v}}_2$, *and* $\sigma^2_{v_{2,j}(\Omega)} = \sigma_2^2$.
- *(iii) Let* $\mathbf{v}$ *be the eigenvector with the largest eigenvalue of the expected alignment matrix* $\bar{A}$*. Then,* $\mathbf{v}(\Omega) \cdot \mathbf{v} \ge 1 - \gamma$, *where* γ *is a small positive number, w.h.p.*
- *(iv)* $1 - \gamma \le \|\bar{\mathbf{v}}\| \le 1$.
- *(v)* $\sigma_1^2 \le \frac{2\gamma}{n}$ *and,* $\sigma_2^2 \le \frac{2\gamma}{(k-1)n}$.
- *(vi) If* $v_1^*/v_2^* = \beta$*, for sufficiently large* n *and w.h.p.,* $\bar{\mathbf{v}}_1 = \frac{a_1}{\sqrt{n}}$ *and* $\bar{\mathbf{v}}_2 = \frac{a_2}{\sqrt{n}}$*, where* $a_1 > a_2$ *and,*

$$\begin{aligned} a_1 &\approx \frac{\beta(1-\gamma)}{\sqrt{\beta^2 + k - 1}}, \\ a_2 &\approx \frac{(1-\gamma)}{\sqrt{\beta^2 + k - 1}}. \end{aligned} \tag{8.18}$$

Proof Owing to symmetry of the problem, for all $1 \le i \le n$, random variables $v_{1,i}(\Omega)$ have the same probability distribution. Therefore, they have the same mean and variance. The same argument holds for random variables $v_{2,j}(\Omega)$ for all $1 \le j \le (k-1)n$. This proves parts (i) and (ii).

To prove part (iii), first we show that, with high probability and for sufficiently large n, $\|E\| \triangleq \|A(\Omega) - \bar{A}\| < \delta n$, where δ is a small positive number, and $\|.\|$ is the spectral norm operator. Since $\bar{A}$ and $A(\Omega)$ have the same diagonal elements, all diagonal elements of E are zero.

Theorem 7 (Gershgorin Circle Theorem) *Let* E *be a complex matrix with entries* $e_{i,j}$*. Let* $R_i = \sum_{j \ne i} |e_{i,j}|$ *be the sum of absolute values of the off-diagonal entries in the row* i*. Let* $D(e_{i,i}, R_i)$ *be the closed disc centered at* $e_{i,i}$ *with radius* R_i*. Every eigenvalue of* E *lies within at least one of the Gershgorin discs* $D(e_{ii}, R_i)$*.*

Proof See reference [76]. ■

We use Gershgorin Circle Theorem to show that, $\|E\| < \delta n$ for sufficiently large n, w.h.p. First, we show that, for sufficiently large n and w.h.p., $R_i < \delta n$, where δ is a small positive number. To simplify notations, we write R_t for different t as follows:

If $i \leq n$, then,

$$R_t = \sum_{i=2}^{n} (\alpha+1)G_i^2 - 2G_i + 1 + \sum_{(i,j)\in\mathcal{B}} (\alpha+1)G_iG_j - G_i - G_j + 1 \\ - (n-1)\big((\alpha+1)p - 2p + 1\big) - (k-1)n\big((\alpha+1)p^2 - 2p + 1\big), \tag{8.19}$$

where G_i is an iid Bernoulli variable with $Pr[G_i = 1] = p$, and $\mathcal{B} = \mathcal{R} - \{(i,i') : 1 \leq i \leq n\}$. Similarly, if $n < t \leq kn$, we can write,

$$R_t = \sum_{(i,j)\in\mathcal{B}} \big((\alpha+1)G_iG_j - G_i - G_j + 1\big) - (k-1)n\big((\alpha+1)p^2 - 2p + 1\big). \tag{8.20}$$

Using Chernoff bound, for sufficiently large n, for a given $\delta_1 > 0$, there exists $\epsilon_1 > 0$ so that,

$$Pr\big[|\frac{1}{n}\sum_{i=1}^{n} G_i - p| > \delta_1\big] \leq e^{-n\epsilon_1}, \tag{8.21}$$
$$Pr\big[|\frac{1}{n}\sum_{i=1}^{n} G_i^2 - p| > \delta_1\big] \leq e^{-n\epsilon_1},$$
$$Pr\big[|\frac{1}{(k-1)n}\sum_{(i,j)\in\mathcal{B}} G_iG_j - p^2| > \delta_1\big] \leq e^{-n\epsilon_1}.$$

Proposition 3 *Let U_1 and U_2 be two random variables such that,*

$$Pr[U_1 \in (-\delta_1, \delta_1)] > 1 - e^{-n\epsilon_1},$$
$$Pr[U_2 \in (-\delta_2, \delta_2)] > 1 - e^{-n\epsilon_2}.$$

Then, w.h.p.,

$$Pr[U_1 + U_2 \geq \delta_1 + \delta_2] \leq e^{-n\min(\epsilon_1,\epsilon_2)},$$
$$Pr[U_1U_2 \geq \delta_1\delta_2] \leq e^{-n\min(\epsilon_1,\epsilon_2)}.$$

Proof Let T be a random variable representing the event $U_1 \in (-\delta_1, \delta_1)$, and T^c be its complement. Then,

$$\begin{aligned} Pr[U_1 + U_2 \geq \delta_1 + \delta_2] =& Pr[U_1 + U_2 \geq \delta_1 + \delta_2 | T]\ Pr[T] \\ &+ Pr[U_1 + U_2 \geq \delta_1 + \delta_2 | T^c]\ Pr[T^c] \\ \leq& Pr[U_2 \geq \delta_2] + Pr[T^c] \\ \leq& e^{-n\min(\epsilon_1,\epsilon_2)}. \end{aligned}$$

A similar argument can be made for the case of U_1U_2. This completes the proof of Proposition 3. ■

According to equations (8.19) and (8.20), for all $1 \leq t \leq kn$, $E[R_t] = 0$. Using Proposition 3 and equation (8.21), there exists $\delta > 0$ such that, $|R_t| \leq \delta n$ for sufficiently large n, w.h.p. Thus, using Gershgorin circle Theorem 7, the maximum eigenvalue of matrix E is smaller than δn, for sufficiently large n, w.h.p., which indicates that $\|E\| \leq \delta n$, for sufficiently large n, w.h.p.

Theorem 8 (Wedin Sin Theorem) *Let* $\mathbf{v}$ *and* $\mathbf{v}(\Omega)$ *be eigenvectors of matrices* $\bar{A}$ *and* $A(\Omega)$ *corresponding to their largest eigenvalues, respectively. Let* λ_1 *and* λ_2 *be the largest and second largest eigenvalues of matrix* $\bar{A}$*. Then, there exists a positive constant* μ *such that,*

$$|\sin \angle(\mathbf{v}, \mathbf{v}(\Omega))| \leq \mu \frac{\|A - A(\Omega)\|}{\lambda_1 - \lambda_2}. \tag{8.22}$$

Proof See reference [77]. ■

According to equation (8.7), as $n \to \infty$, the two largest eigenvalues of the matrix $\bar{A}$ are the roots of equation (8.10) because other eigenvalues are equal to $-a + 1 + \epsilon$ or $-b + 1 + \epsilon$. Solving equation (8.10) for sufficiently large n, we have,

$$\begin{aligned} |\lambda_1 - \lambda_2| &= \sqrt{(\lambda_a - \lambda_b)^2 + 4(k-1)b^2n^2} \\ &= nb\sqrt{(\frac{a}{b} - k + 1)^2 + 4(k-1)} \\ &> nb\sqrt{(1 + \Delta\frac{k-1}{1+\Delta}) + 4(k-1)} \\ &= \frac{b(\beta^2 + k - 1)}{\beta} n. \end{aligned} \tag{8.23}$$

Therefore, using equation (8.23) in Wedin sin Theorem 8, for any small positive δ, we have,

$$|\sin \angle(\mathbf{v}, \mathbf{v}(\Omega))| \leq \frac{\mu\beta}{b(\beta^2 + k - 1)}\delta. \tag{8.24}$$

This completes the proof of part (iii).
To prove part (iv), first we use Jensen's inequality. Since norm is a convex function, we have,

$$\|\bar{\mathbf{v}}\| = \|\mathbb{E}[\mathbf{v}(\Omega)]\| \leq \mathbb{E}\big[\|\mathbf{v}(\Omega)\|\big] = 1. \tag{8.25}$$

From part (iii), we have,

$$\mathbf{v}(\Omega) \cdot \mathbf{v} \geq 1 - \gamma \Rightarrow \bar{\mathbf{v}} \cdot \mathbf{v} \geq 1 - \gamma. \tag{8.26}$$

Then, using Cauchy Schwarz inequality, we have,

$$\|\bar{\mathbf{v}}\| = \|\bar{\mathbf{v}}\|\|\mathbf{v}\| \geq \bar{\mathbf{v}} \cdot \mathbf{v} \geq 1 - \gamma. \tag{8.27}$$

This completes the proof of part (iv).
To prove part (v), we can write,

$$\begin{aligned} \mathbb{E}\big[\|\mathbf{v}(\Omega) - \bar{\mathbf{v}}\|^2\big] = \mathbb{E}\big[1 + \|\bar{\mathbf{v}}\|^2 - 2\mathbf{v}(\Omega) \cdot \bar{\mathbf{v}}\big] &= 1 - \|\bar{\mathbf{v}}\|^2 \\ &\leq 1 - (1-\gamma)^2 = \gamma(2-\gamma) \leq 2\gamma. \end{aligned} \tag{8.28}$$

On the other hand,

$$\mathbb{E}\big[\|\mathbf{v}(\Omega)-\bar{\mathbf{v}}\|^2\big] = n\sigma_1^2 + (k-1)n\sigma_2^2. \tag{8.29}$$

Therefore,

$$\begin{aligned} n\sigma_1^2 \le 2\gamma &\Rightarrow \sigma_1^2 \le \frac{2\gamma}{n}, \\ (k-1)n\sigma_2^2 \le 2\gamma &\Rightarrow \sigma_2^2 \le \frac{2\gamma}{(k-1)n}. \end{aligned} \tag{8.30}$$

This completes the proof of part (v).

To prove part (vi), we assume that in the worst case, vectors $\mathbf{v}$ and $\bar{\mathbf{v}}$ has inner product value of $1-\gamma$. Therefore, using part (iii) and (iv), we have,

$$\begin{cases} n v_1^* \bar{\mathbf{v}}_1 + (k-1)n v_2^* \bar{\mathbf{v}}_2 = 1-\gamma, \\ n\bar{\mathbf{v}}_1^2 + (k-1)n\bar{\mathbf{v}}_2^2 = c, \end{cases} \tag{8.31}$$

where $1-\gamma \le c = \|\bar{\mathbf{v}}\| \le 1$. Solving equation (8.31) using equation (8.17) for sufficiently large n, we have,

$$\begin{aligned} a_1 &= \frac{\beta(1-\gamma) + \sqrt{(k-1)(c-(\gamma-1)^2)}}{\sqrt{\beta^2+k-1}}, \\ a_2 &= \frac{(1-\gamma)\sqrt{k-1} - \beta\sqrt{c-(\gamma-1)^2}}{\sqrt{(k-1)(\beta^2+k-1)}}. \end{aligned} \tag{8.32}$$

Choosing δ sufficiently small and n sufficiently large, equation (8.32) leads to equation (8.31). This completes the proof of part (vi). ■

Now, we have all technology to prove Theorem 1. Recall notations $\mathcal{S}_1 = \{1, 2, \ldots, n\}$ and $\mathcal{S}_2 = \{n+1, n+2, \ldots, kn\}$. Let $\mathcal{S}_1(m) \subseteq \mathcal{S}_1$ and $\mathcal{S}_2(m) \subseteq \mathcal{S}_2$ where $|\mathcal{S}_1(m)| = |\mathcal{S}_2(m)| = m$. Define following variables:

$$\begin{aligned} U_1 &\triangleq \frac{1}{n}\sum_{i\in\mathcal{S}_1} v_i(\Omega), \\ U_2 &\triangleq \frac{1}{n}\Big[\sum_{i\in\mathcal{S}_1(n-m)} v_i(\Omega) + \sum_{j\in\mathcal{S}_2(m)} v_j(\Omega)\Big]. \end{aligned} \tag{8.33}$$

Let $\rho = m/n \neq 0$. According to Lemma 4, $\mathbb{E}[U_1] = \bar{\mathbf{v}}_1$ and $\mathbb{E}[U_2] = (1-\rho)\bar{\mathbf{v}}_1 + \rho\bar{\mathbf{v}}_2$. Moreover, $\sigma_{U_1}^2 \le \sigma_1^2$ and $\sigma_{U_2}^2 \le (1-\rho)\sigma_1^2 + \rho\sigma_2^2 < \sigma_1^2$. Define,

$$d \triangleq \frac{\big|\mathbb{E}[U_1] - \mathbb{E}[U_2]\big|}{2} = \frac{c_1(1-\gamma)}{\sqrt{n}}, \tag{8.34}$$

where $c_1 = \frac{\rho\Delta}{2\sqrt{(1+\Delta)^2+k-1}}$.

Using Chebyshev's inequality, we have,

$$Pr[U_1 \le \mathbb{E}[U_1] - d] \le \frac{2}{c_1^2}\frac{\gamma}{(1-\gamma)^2} \le c_2\gamma = \epsilon_1, \tag{8.35}$$
$$Pr[U_2 \ge \mathbb{E}[U_2] + d] \le \frac{2}{c_1^2}\frac{\gamma}{(1-\gamma)^2} \le c_2\gamma = \epsilon_1.$$

Therefore,

$$Pr[U_1 < U_2] < \epsilon_1, \tag{8.36}$$

where ϵ_1 can be arbitrarily small for sufficiently large n. This completes the proof of Theorem 1.

8.2 Proof of Theorem 2

Without loss of generality and similarly to the proof of Theorem 1, let $P = I$. Let A be the alignment network of graphs G_1 and G_2 defined according to equation (2.3). Similarly to the proof of Theorem 1, re-order 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 network $\bar{A}$ 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 networks. Recall notations $\mathcal{S}_1 = \{1, 2, \ldots, n\}$ and $\mathcal{S}_2 = \{n+1, n+2, \ldots, kn\}$.

First, we consider the noise model I (3.8):

Define,

$$\begin{aligned} a' \triangleq& p(1-p_e)(\alpha+\epsilon) + (1-p)(1-p_e)(1+\epsilon) + (pp_e + (1-p)p_e)\epsilon \\ b' \triangleq& \big(p^2(1-p_e) + pp_e(1-p)\big)(\alpha+\epsilon) \\ &+\big((1-p)^2(1-p_e) + pp_e(1-p)\big)(1+\epsilon) \\ &+\big(2p(1-p)(1-p_e) + 2p^2p_e\big)\epsilon. \end{aligned} \tag{8.37}$$

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

$$\bar{A}(t_1, t_2) = \begin{cases} a', & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \ne t_2, \\ b', & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \ne t_2, \\ 1+\epsilon, & \text{if } t_1 = t_2. \end{cases} \tag{8.38}$$

We have,

$$a' - b' = (\alpha+1)(2p_e - 1)p(p-1) + p_e(1-2p)\epsilon. \tag{8.39}$$

Thus, if $p \ne 0, 1$ 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 (3.9) is similar. To simplify notation and illustrate the main idea, here we assume ϵ is sufficiently small with negligible effects.

Define,

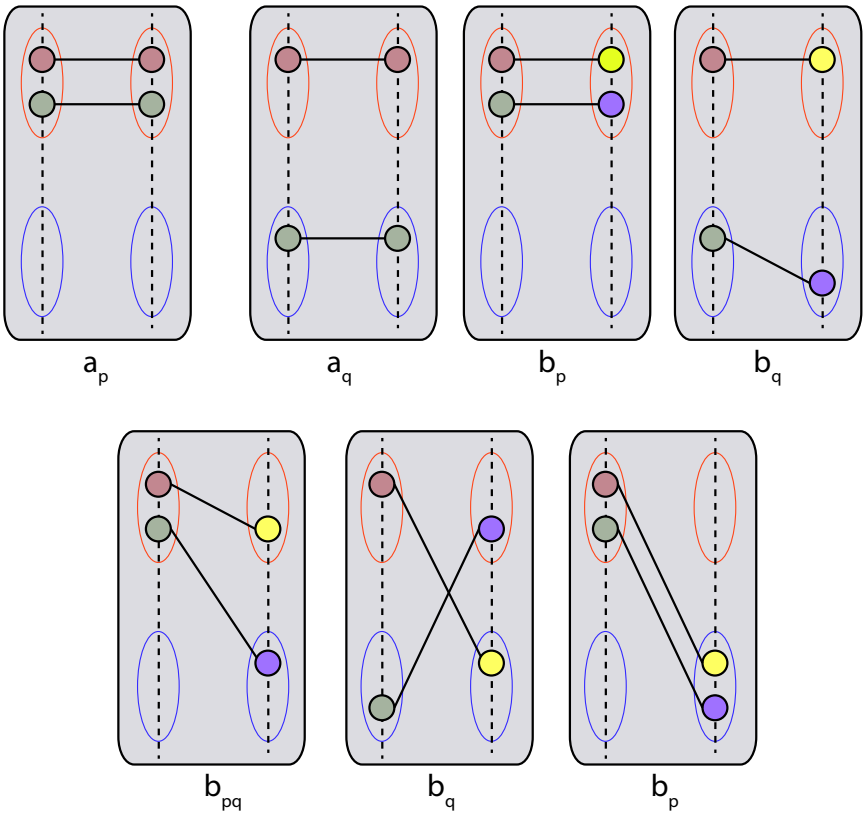

Figure 21: Expected alignment scores of different mapping combinations. Ellipses with the same color represent corresponding modules across two networks. Nodes with the same color represent true mappings across two networks.

$$
\begin{aligned}
a'' \triangleq & p(1-p_e)(\alpha) + (1-p)(1-p_{e_2}) = 1 - p\big(1 + \alpha(p_e - 1) + p_e\big) \\
b'' \triangleq & p^2(1-p_e)\alpha + (1-p)^2(1-p_{e_2}) + 2p(1-p)p_{e_2}(1+\alpha) \\
= & 1 - p(2 + p_e) + p^2(1 + \alpha + 2p_e).
\end{aligned} \tag{8.40}
$$

The expected alignment network in this case is:

$$
\bar{A}(t_1, t_2) = \begin{cases} a'', & \text{if } t_1 \text{ and } t_2 \in \mathcal{S}_1, t_1 \neq t_2, \\ b'', & \text{if } t_1 \text{ or } t_2 \in \mathcal{S}_2, t_1 \neq t_2, \\ 1 + \epsilon, & \text{if } t_1 = t_2. \end{cases} \tag{8.41}
$$

Moreover, we have,

$$
a'' - b'' = p\big((1 - p - p_e)(1 + \alpha) + p_e(1 - 2p)\big). \tag{8.42}
$$

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

8.3 Proof of Theorem 3

Without loss of generality, let $P = I$ in (3.11). Let A be the alignment network of graphs G_1 and G_2 defined according to (2.3). G_1 and G_2 are stochastic block networks with m modules, each with n nodes. Modules of G_1 are represented by $\{V_1^1, V_1^2, \ldots, V_1^m\}$ where $V_1^a = \{i_a^1, i_a^2, \ldots, i_a^n\}$. Similarly, modules of G_2 are represented by $\{V_2^1, V_2^2, \ldots, V_2^m\}$ where $V_2^a = \{j_a^1, j_a^2, \ldots, j_a^n\}$. The alignment network A has n^2m^2 nodes. We re-order row (and column) indices of matrix A so that the first mn^2 indices are within module mappings:

$$(i_1^1, j_1^1), (i_1^2, j_1^2), ..., (i_1^n, j_1^n), (i_1^1, j_1^2), (i_1^1, j_1^3), ..., (i_1^n, j_1^{n-1}).$$

The remaining $(m^2 - m)n^2$ indices correspond to across module mappings:

$$(i_1^1, j_2^1), (i_1^1, j_2^2), ..., (i_m^n, j_{m-1}^{n-1}), (i_m^n, j_{m-1}^n).$$

Define,

$$\begin{aligned} a_p &\triangleq (\alpha - 1)p + 1, \\ b_p &\triangleq (\alpha + 1)p^2 - 2p + 1, \\ a_q &\triangleq (\alpha - 1)q + 1, \\ b_q &\triangleq (\alpha + 1)q^2 - 2q + 1, \\ b_{pq} &\triangleq \alpha pq + (1 - p)(1 - q). \end{aligned} \tag{8.43}$$

To form the expected alignment network $\bar{A}$, we consider seven types of mapping-pairs illustrated in Figure 21. Thus, the expected network alignment matrix has the following structure:

$$\bar{A} = \left[\begin{array}{ccccc|ccccc} A_1 & A_2 & A_2 & \cdots & A_2 & A_3 & & \cdots & & A_3 \\ A_2 & A_1 & A_2 & \cdots & A_2 & & & & & \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ A_2 & & \cdots & & A_1 & A_3 & & \cdots & & A_3 \\ \hline A_3 & & \cdots & & A_3 & A_4 & A_5 & A_5 & \cdots & A_5 \\ & & & & & A_5 & A_4 & A_5 & \cdots & A_5 \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ A_3 & & \cdots & & A_3 & A_5 & & \cdots & & A_4 \end{array}\right], \tag{8.44}$$

where the upper and lower blocks have mn^2 and $(m^2 - m)n^2$ rows, respectively. A_i is a matrix of size $n^2 \times n^2$, defined as follows:

$$A_1 = \left[\begin{array}{ccc|ccc} a_p & \cdots & a_p & b_p & \cdots & b_p \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_p & \cdots & a_p & b_p & \cdots & b_p \\ \hline b_p & \cdots & b_p & b_p & \cdots & b_p \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ b_p & \cdots & b_p & b_p & \cdots & b_p \end{array}\right], \tag{8.45}$$

$$A_2 = \left[\begin{array}{ccc|ccc} a_q & \cdots & a_q & b_q & \cdots & b_q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_q & \cdots & a_q & b_q & \cdots & b_q \\ \hline b_q & \cdots & b_q & b_q & \cdots & b_q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ b_q & \cdots & b_q & b_q & \cdots & b_q \end{array}\right],$$

$$A_3 = b_{pq}\mathbb{1}_{n^2},$$

$$A_4 = b_p\mathbb{1}_{n^2},$$

$$A_5 = b_q\mathbb{1}_{n^2},$$

where $\mathbb{1}_{n^2}$ is a $n^2 \times n^2$ matrix whose elements are ones.

Suppose v is the leading eigenvector of the expected alignment matrix $\bar{A}$. We wish to show that,

$$v(i) > v(j), \quad \forall 1 \le i \le mn^2, \forall mn^2 < j \le m^2n^2. \tag{8.46}$$

Lemma 5 *Let A be a positive matrix. If one or any number of entries of row i are increased and all the other rows remain fixed, and if the i-th entry of the Perron vector is held a fixed constant equal to 1, then the remaining entries of the Perron vector strictly decrease.*

Proof See a proof in [78]. ■

Define matrix B as follows:

$$B = \left[\begin{array}{ccccc|ccccc} B_1 & B_2 & B_2 & \cdots & B_2 & B_3 & & \cdots & & B_3 \\ B_2 & B_1 & B_2 & \cdots & B_2 & & & & & \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ B_2 & & \cdots & & B_1 & B_3 & & \cdots & & B_3 \\ \hline B_3 & & \cdots & & B_3 & B_1 & B_2 & B_2 & \cdots & B_2 \\ & & & & & B_2 & B_1 & B_2 & \cdots & B_2 \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ B_3 & & \cdots & & B_3 & B_2 & & \cdots & & B_1 \end{array}\right], \tag{8.47}$$

where,

$$B_1 = b_p\mathbb{1}_{n^2}, \tag{8.48}$$

$$B_2 = b_q\mathbb{1}_{n^2},$$

$$B_3 = b_{pq}\mathbb{1}_{n^2}.$$

Because $a_p > b_p$ and $a_q > b_q$, using Lemma 5, it is sufficient to show (8.46) for the matrix B. Suppose v_B is the leading eigenvector of the matrix B. We wish to show,

$$v_B(i) > v_B(j), \quad \forall 1 \le i \le mn^2, \forall mn^2 < j \le m^2n^2. \tag{8.49}$$

Note that,

$$B = \mathbb{1}_{n^2} \otimes C, \tag{8.50}$$

where C is a $m^2 \times m^2$ matrix with a structure illustrated as follows:

$$C = \left[\begin{array}{ccccc|ccccc} b_p & b_q & b_q & \cdots & b_q & b_{pq} & & \cdots & & b_{pq} \\ b_q & b_p & b_q & \cdots & b_q & & & & & \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ b_q & & \cdots & & b_p & b_{pq} & & \cdots & & b_{pq} \\ \hline b_{pq} & & \cdots & & b_{pq} & b_p & b_q & b_q & \cdots & b_q \\ & & & & & b_q & b_p & b_q & \cdots & b_q \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ b_{pq} & & \cdots & & b_{pq} & b_q & & \cdots & & b_p \end{array}\right]. \tag{8.51}$$

Let v_C be the leading eigenvector of the matrix C. Using the property of Kronecker products, we have,

$$v_B = \mathbb{1}_{n^2} \otimes v_C. \tag{8.52}$$

Therefore, to show (8.46) and (8.49), it is sufficient to show that,

$$v_C(i) > v_C(j), \quad \forall 1 \le i \le m, \forall m < j \le m^2. \tag{8.53}$$

Lemma 6 *Consider the matrix*

$$X = \left[\begin{array}{ccccc|ccccc} c & b & b & \cdots & b & a & & \cdots & & a \\ b & c & b & \cdots & b & & & & & \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ b & & \cdots & & c & a & & \cdots & & a \\ \hline a & & \cdots & & a & c & b & b & \cdots & b \\ & & & & & b & c & b & \cdots & b \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ a & & \cdots & & a & b & & \cdots & & c \end{array}\right], \tag{8.54}$$

where the size of the top and bottom blocks are n_1 and n_2, respectively. Let v^* be the leading eigenvector of X. Then, if $a > b > 0$, $c > 0$, and $n_2 > n_1$, we have,

$$v^*(i) > v^*(j), \quad \forall 1 \le i \le n_1, \forall n_1 < j \le n_1 + n_2. \tag{8.55}$$

Proof

Suppose v is an eigenvector of the matrix X with the corresponding eigenvalue λ. Owing to the symmetric structure of the matrix X, we have,

$$\begin{cases} v(1) = ... = v(n_1) \triangleq v_1, \\ v(n_1+1) = ... = v(n_1+n_2) \triangleq v_2. \end{cases} \tag{8.56}$$

Using (8.56) in the eigen decomposition equality, we have,

$$\begin{cases} cv_1 + (n_1-1)bv_1 + n_2 a v_2 = \lambda v_1 \\ n_1 a v_1 + cv_2 + (n_2-1)bv_2 = \lambda v_2. \end{cases} \tag{8.57}$$

Thus, we have,

$$(\lambda' - \lambda_1)(\lambda' - \lambda_2) = n_1 n_2 a^2, \tag{8.58}$$

where,

$$\begin{cases} \lambda' = \lambda - c, \\ \lambda_1 = (n_1-1)b, \\ \lambda_2 = (n_2-1)b. \end{cases} \tag{8.59}$$

Let λ^* be the largest root of (8.58), which corresponds to the leading eigenvector of the matrix v^*. To prove the lemma, it is sufficient to show that,

$$\lambda^* > n_1 a + (n_2-1)b. \tag{8.60}$$

This is true if $a > b$ and $n_2 > n_1$. To show this, we need to show,

$$\lambda^* = \frac{\lambda_1 + \lambda_2 + \sqrt{(\lambda_1 - \lambda_2)^2 + 4n_1 n_2 a^2}}{2} > n_1 a + (n_2-1)b, \tag{8.61}$$

which is true under the conditions of the lemma. This completes the proof. ■

If $p > q$ and $\alpha > 1/q - 1$, we have $b_{pq} > b_q$. Thus, Lemma 6 leads to (8.53). This completes the proof.

8.4 Proof of Theorem 4

We use the same setup considered in the proof of Theorem 3 to form the expected alignment network. Considering $0 < p_e^2 \ll 1$ and $0 < p_{e_2}^2 \ll 1$, expected scores of different mapping-pairs illustrated in Figure 21 can be approximated as follows:

$$\begin{aligned} a_p &\simeq p(1-p_e)\alpha, \\ b_p &\simeq p^2(1-p_e)^2\alpha + 2p(1-p)p_{e_2}\alpha, \\ a_q &\simeq q(1-p_e)\alpha, \\ b_q &\simeq q^2(1-p_e)^2\alpha + 2q(1-q)p_{e_2}\alpha, \\ b_{pq} &\simeq pq(1-p_e)^2\alpha + p(1-q)p_{e_2}\alpha + (1-p)qp_e\alpha. \end{aligned} \tag{8.62}$$

The proof is similar to the one of Theorem 3. To use Lemma 5, we need to have,

$$\begin{cases} a_p > b_p, \\ a_q > b_q, \end{cases} \tag{8.63}$$

which results in following conditions:

$$\begin{aligned} p &< \frac{1 - pp_e}{1 + p_e^2}, \\ q &< \frac{1 - pp_e}{1 + p_e^2}. \end{aligned} \tag{8.64}$$

Because $p_e^2 << 1$, we then have,

$$\begin{aligned} p &< \frac{1}{1 + p_e}, \\ q &< \frac{1}{1 + p_e}. \end{aligned} \tag{8.65}$$

To use Lemma 6, we need to have $b_{pq} > b_q$. Using (8.62), we have:

$$b_{pq} - b_q = (1-p)q(p-q) + p_e\Big(\underbrace{p^2(1+2q) - 6pq + q(1+2q)}_{\text{polynomial I}}\Big). \tag{8.66}$$

To show the non-negativity of the right-hand side of (8.66), it is sufficient to show the non-negativity of polynomial I. This polynomial has two roots at

$$p_{roots} = \frac{6q \pm \sqrt{-4q(4q-1)(q-1)}}{2(1+2q)}. \tag{8.67}$$

If $0 < q \leq 1/4$,

$$-4q(4q-1)(q-1) < 0. \tag{8.68}$$

Because the value of the polynomial I at $p = 0$ is positive, if $0 < q \leq 1/4$, the polynomial is always non-negative. If $q > 1/4$, we need to have,

$$p \leq \frac{6q - \sqrt{4q(4q-1)(1-q)}}{2(1+2q)}, \tag{8.69}$$

which guarantees the non-negativity of polynomial I. The rest of the proof is similar to the one of Theorem 3.

8.5 Proof of Theorem 5

We use the same setup considered in the proof of Theorem 3 to form the expected alignment network. Suppose $S(P)$ and $S(\tilde{P})$ correspond to the expected objective function of the network alignment optimization 2.5 using permutation matrices P and $\tilde{P}$, respectively, where,

$$\frac{1}{nm}\|P - \tilde{P}\| > 0. \tag{8.70}$$

We wish to show that, $S(P) > S(\tilde{P})$. We have,

$$\begin{aligned} S(P) =& mn^2 a_p + (m^2 - m)n^2 a_q \\ >& (mn)^2 a_q \end{aligned} \tag{8.71}$$

where a_p and a_q are defined according to (8.43). Under conditions of Theorem 3, we have $b_p > b_{pq} > b_q$ according to (8.43). Thus,

$$S(\tilde{P}) \leq (mn)^2 b_p. \tag{8.72}$$

Using (8.71) and (8.72), we need to show that $a_q > b_p$. We have,

$$b_p - a_q = (\alpha + 1)p^2 - 2p - (\alpha - 1)q. \tag{8.73}$$

This polynomial have two roots at

$$p_{roots} = \frac{1 \pm \sqrt{1 + (\alpha^2 - 1)q}}{\alpha + 1}. \tag{8.74}$$

Because $\alpha > 1$, the minimum root is always negative. Moreover, at $p = 0$, the polynomial value is negative. Thus, (8.73) is negative if

$$0 < p \leq \frac{1 + \sqrt{1 + (\alpha^2 - 1)q}}{1 + \alpha}. \tag{8.75}$$

This completes the proof.

8.6 Proof of Theorem 6

We use the same setup considered in the proof of Theorem 3 to form the expected alignment network. Similarly to the proof of Theorem 5, we need to show $a_q > b_p$ according to (8.62). We have,

$$a_q - b_p = \alpha\big(q(1 - p_e) - p^2(1 + p_e^2)\big) \tag{8.76}$$

which is positive if

$$p^2 \leq \frac{q(1 - p_e)}{1 + p_e^2}. \tag{8.77}$$

This completes the proof.

9 Conclusion and Future Directions

We introduced a network alignment algorithm called *EigenAlign* which aims to find a bijective mapping across vertices of two graphs to maximize the number of overlapping edges and to minimize the number of mismatched interactions across networks. EigenAlign creates a simple relaxation for the underlying QAP by relaxing binary assignment constraints linearly along the leading eigenvector of the alignment matrix. This leads to an eigenvector solution for the underlying network alignment optimization which can be solved efficiently through an eigen decomposition step followed by a linear assignment step. Unlike existent network alignment methods, EigenAlign considers both matched and mismatched interactions in its optimization and therefore, it is effective in aligning networks even with low similarity. This is critical in comparative analysis of biological networks of distal species because there are numerous mismatched interactions across those networks partially owing to extensive gene functional divergence due to processes such as gene duplication and loss.

For Erdös-Rényi graphs graphs, we proved that, the EigenAlign solution is asymptotically optimal with high probability, under some general conditions. Through simulations, we compared the performance of the EigenAlign algorithm with the one of existent network alignment methods based on belief propagation (NetAlign), spectral decomposition (IsoRank), Lagrange relaxation (Klau optimization), and a SDP-based method. Our simulations illustrated the effectiveness of the EigenAlign algorithm in aligning various network structures such as Erdös-Rényi, power law, and stochastic block structures, under different noise models.

For modular network structures, we showed that, EigenAlign can be used to split the large QAP into small subproblems. This enables the use of computationally expensive, but tight, semidefinite programming relaxations over each subproblem. We termed this hybrid method *EigenAlign+SDP*, which has high performance and low computational complexity. Note that, gene regulatory networks do not have the stochastic block structure considered in this method. To be able to use a SDP-based relaxation in our biological experiments, we applied the SDP relaxation to align small regulatory sub-graphs determined by homologous gene families. However, owing to small sizes of these gene families, this method did not perform well in our experiments. Designing practical SDP-based network alignment methods with low computational complexity remains a promising direction for feature work.

Identifying gene regulatory interactions and modules conserved across distal species can shed light on functional regulatory programs and pathways across diverse phylogenies, eventually leading to better understanding of human biology. To that end, we applied EigenAlign to compare gene regulatory networks across human, fly and worm species to infer conservation of individual regulatory connections which can be used to compute conserved pathways and modules across human, fly and worm organisms. EigenAlign inferred conserved regulatory interactions across these species despite large evolutionary distances spanned. Using EigenAlign mappings, we found strong conservation of centrally-connected genes and some biological pathways, especially for human-fly comparisons.

To compare regulatory pathways across human, fly and worm, we inferred regulatory interactions in these species by integrating genome-wide functional and physical regulatory evidences. We found that, these inferred interactions, especially in human and fly, overlap significantly with known benchmarks. These inferred interactions can be used as a guide to design informative high-throughput experiments to infer accurate context-specific regulatory interactions. Moreover, inferred regulatory interactions, specially conserved ones, provide useful resources to understand regulatory roles of individual genes, pathways and modules in human diseases [58], cancer [79] and

drug designs [80–82]. Recently researchers have hypothesized that some human diseases can be due to single nucleotide variants (SNVs) sitting in enhancer-like regions of the genome and are typically enriched in transcription factor binding sites [83]. Therefore, using regulatory networks can help us identifying direct and indirect target genes and higher-order regulatory pathways of these disease-causing SNVs [84]. Moreover, drugs targeting specific conserved regulatory pathways can be tested first in model organism, reducing experimental costs and increasing their efficiency [80–82]. We believe that our inferred regulatory networks and network analysis techniques can make a significant impact in many areas of molecular and cell biology to study complex diseases, drug designs, and beyond.

10 Acknowledgements

Authors thank Mariana Mendoza for early processing of regulatory datasets.

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. i259–1267, 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. Kalaev, 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] P. Schweitzer, "Problems of unknown complexity: graph isomorphism and ramsey theoretic numbers," Ph.D. dissertation, Saarbrucken, Univ., Diss., 2009, 2009.

[14] R. E. Burkard, *Quadratic assignment problems*. Springer, 2013.

[15] G. W. Klau, "A new graph-based method for pairwise global network alignment," *BMC bioinformatics*, vol. 10, no. Suppl 1, p. S59, 2009.

[16] Z. Li, S. Zhang, Y. Wang, X.-S. Zhang, and L. Chen, "Alignment of molecular networks by integer quadratic programming," *Bioinformatics*, vol. 23, no. 13, pp. 1631–1639, 2007.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[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] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] E. L. Lawler, "The quadratic assignment problem," *Management science*, vol. 9, no. 4, pp. 586–599, 1963.

[35] 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.

[36] A. Frieze and J. Yadegar, "On the quadratic assignment problem," *Discrete applied mathematics*, vol. 5, no. 1, pp. 89–98, 1983.

[37] 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.

[38] 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.

[39] M. Carcassoni and E. R. Hancock, "Alignment using spectral clusters." in *BMVC*, 2002, pp. 1–10.

[40] T. Czajka and G. Pandurangan, "Improved random graph isomorphism," *Journal of Discrete Algorithms*, vol. 6, no. 1, pp. 85–92, 2008.

[41] 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.

[42] L. Babai, P. Erdo s, and S. M. Selkow, "Random graph isomorphism," *SIAM Journal on Computing*, vol. 9, no. 3, pp. 628–635, 1980.

[43] D. B. West *et al.*, *Introduction to graph theory.* Prentice hall Upper Saddle River, 2001, vol. 2.

[44] 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.

[45] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[46] G. Evanno, S. Regnaut, and J. Goudet, "Detecting the number of clusters of individuals using the software structure: a simulation study," *Molecular ecology*, vol. 14, no. 8, pp. 2611–2620, 2005.

[47] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," *Psychometrika*, vol. 50, no. 2, pp. 159–179, 1985.

[48] A. Bobbio and K. S. Trivedi, "An aggregation technique for the transient analysis of stiff markov chains," *Computers, IEEE Transactions on*, vol. 100, no. 9, pp. 803–814, 1986.

[49] 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.

[50] 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.

[51] 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.

[52] 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.

[53] 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.

[54] 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.

[55] R. Sharan and T. Ideker, "Modeling cellular machinery through biological network comparison," *Nature biotechnology*, vol. 24, no. 4, pp. 427–433, 2006.

[56] 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.

[57] 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.

[58] 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.

[59] 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.

[60] 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.

[61] 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.

[62] 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.

[63] S. Feizi, D. Marbach, M. Médard, and M. Kellis, "Network deconvolution as a general method to distinguish direct dependencies in networks," *Nature biotechnology*, 2013.

[64] 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.

[65] 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.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] Y.-C. Wu, M. S. Bansal, M. D. Rasmussen, J. Herrero, and M. Kellis, "Phylogenetic identification and functional validation of orthologous genes across human, mouse, fly, worm, yeast," *submitted to Genome Research, available on arXiv*, 2014.

[72] L. d. F. Costa, F. A. Rodrigues, G. Travieso, and P. R. Villas Boas, "Characterization of complex networks: A survey of measurements," *Advances in Physics*, vol. 56, no. 1, pp. 167–242, 2007.

[73] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35–41, 1977.

[74] M. P. Joy, A. Brock, D. E. Ingber, and S. Huang, "High-betweenness proteins in the yeast protein interaction network," *BioMed Research International*, vol. 2005, no. 2, pp. 96–103, 2005.

[75] P. D. Thomas, V. Wood, C. J. Mungall, S. E. Lewis, J. A. Blake, G. O. Consortium *et al.*, "On the use of gene ontology annotations to assess functional similarity among orthologs and paralogs: a short report," *PLoS computational biology*, vol. 8, no. 2, p. e1002386, 2012.

[76] S. A. Gershgorin, "Uber die abgrenzung der eigenwerte einer matrix," . , no. 6, pp. 749–754, 1931.

[77] P.-A. Wedin, "Perturbation bounds in connection with singular value decomposition," *Informationsbehandling (BIT)*, vol. 12, pp. 99–111, 1972.

[78] E. Deutsch and M. Neumann, "On the first and second order derivatives of the perron vector," *Linear algebra and its applications*, vol. 71, pp. 57–76, 1985.

[79] J. Frasor, J. M. Danes, B. Komm, K. C. Chang, C. R. Lyttle, and B. S. Katzenellenbogen, "Profiling of estrogen up-and down-regulated gene expression in human breast cancer cells: insights into gene networks and pathways underlying estrogenic control of proliferation and cell phenotype," *Endocrinology*, vol. 144, no. 10, pp. 4562–4574, 2003.

[80] A. L. Hopkins, "Network pharmacology: the next paradigm in drug discovery," *Nature chemical biology*, vol. 4, no. 11, pp. 682–690, 2008.

[81] L. Xie, J. Li, L. Xie, and P. E. Bourne, "Drug discovery using chemical systems biology: identification of the protein-ligand binding network to explain the side effects of cetp inhibitors," *PLoS computational biology*, vol. 5, no. 5, p. e1000387, 2009.

[82] P. Csermely, V. Agoston, and S. Pongor, "The efficiency of multi-target drugs: the network approach might help drug design," *Trends in Pharmacological Sciences*, vol. 26, no. 4, pp. 178–182, 2005.

[83] M. T. Maurano, R. Humbert, E. Rynes, R. E. Thurman, E. Haugen, H. Wang, A. P. Reynolds, R. Sandstrom, H. Qu, J. Brody *et al.*, "Systematic localization of common disease-associated variation in regulatory dna," *Science*, vol. 337, no. 6099, pp. 1190–1195, 2012.

[84] A. B. Glinskii, J. Ma, S. Ma, D. Grant, C.-U. Lim, S. Sell, and G. V. Glinsky, "Identification of intergenic trans-regulatory rnas containing a disease-linked snp sequence and targeting cell cycle progression/differentiation pathways in multiple common human disorders," *Cell Cycle*, vol. 8, no. 23, pp. 3925–3942, 2009.