Primal-Dual Analysis for Scalable Soft Network Alignment

Yifan Xu
Department of Computer Science
Rice University
Houston, TX 77005
yx76@rice.edu

Abstract

Network alignment is a fundamental problem in various domains, including bioinformatics, social network analysis, computer vision, and computational ecology. This report investigates the soft network alignment problem from an optimization perspective, proposing an entropically regularized formulation that enables efficient gradient-based optimization. Leveraging primal-dual methods, we derive explicit gradient and Hessian formulations to facilitate scalable and accurate solutions. Through experimental evaluations on both synthetic toy graphs and real-world ecological networks, we demonstrate that soft alignments provide richer structural insights compared to traditional hard alignments. These findings highlight the advantages of soft alignment methods in capturing nuanced structural similarities, making them particularly beneficial when dealing with complex or large-scale network alignment tasks.

1 Introduction

1.1 Background and Motivation

The *network alignment problem* seeks a mapping between the vertex sets of two graphs that aligns or “matches" similar nodes. This problem is of interest in various settings:

- **Bioinformatics:** Aligning protein-protein interaction networks or gene regulatory networks to detect similar functional modules [10]
- **Social Network Analysis:** Matching user accounts across different social platforms [3]
- **Computer Vision:** Aligning graphs that represent visual features in multimodal detection [13]
- **Computational Ecology:** Aligning food webs to identify common structural backbones [1]

Generally, given two graphs $G_1 = (V_1, E_1, w_1)$ and $G_2 = (V_2, E_2, w_2)$, the graph alignment problem seeks a matching $\mathcal{M} \subseteq V_1 \times V_2 \times \mathbb{R}$, such that all matchings $(u, v, w) \in \mathcal{M}$ align node $u \in V_1$ with $v \in V_2$ with some weight $w \in \mathbb{R}$. We have two general formulations of the network alignment problem:

1. *Hard alignment problem:* the weights are binary, i.e., $w_k \in \{0, 1\} \quad \forall k \in [|\mathcal{M}|]$ and each node must align to exactly one node from another graph if possible;
2. *Soft (fractional/distributional) alignment problem:* the weights are nonnegative, i.e., $w_k \geq 0 \quad \forall k \in [|\mathcal{M}|]$, and the sum of the weights of all matchings incident to any node is consistent over each graph.

Preprint. Under review.

1.2 Feature Vector Embedding

Often, matchings are desired to satisfy some heuristic, e.g., “structural similarity". To quantify such heuristics, it is common practice to create embedding vectors for each node of the two graphs[12]. We can then compute the heuristic by defining a metric $d(\cdot, \cdot)$ in the feature space. More formally, we create a mapping $\phi : V_1 \cup V_2 \to \mathbb{R}^p$ which captures the essential characteristics in the application scenario. Notably, [5] creates such a mapping through random walks that maximizes the likelihood of preserving the neighborhood of nodes to capture local structural information. Computational ecologists proposed network motifs[9], where each component of the feature vector contains the count for the number of times the node participates in certain sub-structures. Various other approaches exist in the literature[2], but generally we work with the projection of the vertices in the feature space(i.e., $\phi(V_1)$ and $\phi(V_2)$) during network alignment.

This report restates the hard and soft alignment problems from an optimization perspective. To do this, we attempt to minimize the sum of the cost for the alignment across the entire space of alignments. Moreover, to avoid the trivial solution (nothing aligning to nothing, i.e., $\mathcal{M} = \emptyset$) always being optimal, we introduce a non-alignment penalty ϵ, so that each node that is aligned with another node(as opposed to aligning to nothing) reduces the cost function by ϵ. Then, we discuss various optimization tools(the Hungarian method[7], the Gurobi solver[6], dual ascent, and various accelerated methods[4]) that we can leverage to solve these problems efficiently and accurately.

2 Problem Formulation

2.1 Notations and Definitions

Let

$$G_1 = (V_1, E_1), \quad G_2 = (V_2, E_2)$$

with $|V_1| = m$ and $|V_2| = n$. We list the nodes in G_1 as $u_1, u_2, \cdots, u_m$ and those in G_2 as $v_1, v_2, \cdots, v_n$. As discussed in section 1.2, each node u_i or v_j is associated with a feature vector via the embedding:

$$\phi : V_1 \cup V_2 \to \mathbb{R}^p$$

Let $\Delta_k := \{p \in \mathbb{R}^k : p_i \geq 0, \sum_{i=1}^k p_i = 1\}$ denote the probability simplex of dimension k. Let $[k] := \{1, 2, \cdots, k\}$ denote the positive integers up to k (inclusive). Further, define $d(\cdot, \cdot)$ to be a nonnegative cost function on $\mathbb{R}^p \times \mathbb{R}^p$.

For each alignment $\mathcal{M}$ and fixed ordering of the nodes $u_1, u_2, \cdots, u_m$ and $v_1, v_2, \cdots, v_n$, we define an equivalent *alignment matrix* T as follows:

Definition 2.0.1 (Alignment Matrix) *An alignment matrix T corresponding to the alignment $\mathcal{M}$ is a $|V_1| \times |V_2|$ matrix such that for all $i \in \{1, ..., m\}$ and $j \in \{1, ..., n\}$*

1. $[T]_{ij} = w_{u_i, v_j} \in [0, 1]$
2. $[T\mathbb{1}]_i \leq 1$
3. $[T^\top \mathbb{1}]_j \leq 1$

In the case of hard alignments, we can write further that $[T]_{ij} \in \{0, 1\}$.

2.2 Alignment Formulations

Definition 2.0.2 (Hard Alignment Problem) *A hard alignment between G_1 and G_2, is a set $\mathcal{M}$ where*

$$\mathcal{M} \subseteq (V_1 \cup \emptyset) \times (V_2 \cup \emptyset)$$

, and

$$w_{\cdot,\cdot} \in \{0, 1\}$$

such that no node in either V_1 and V_2 can belong to more than one pair.

Notice that in this definition, we allow for an alignment to nothing(i.e., $(u_i, \emptyset)$ or $(\emptyset, v_j)$). This enables us to align networks of different sizes. However, this also gives rise to a problem: what cost should we assign to such matchings? One solution would be to assign an incentive $\epsilon > 0$ for every alignment between nodes. Notice that ϵ should be comparable in order of magnitude to the average of the costs $d(\phi(u_i), \phi(v_j))$. Then, the above problem can be formulated as the following binary integer program:

Definition 2.0.3 (Hard Alignment Optimization Formulation #1)

$$\begin{array}{ll} \mathit{min}_{T \in \mathbb{R}^{m \times n}} & \sum_{i=1}^{m} \sum_{j=1}^{n} d(\phi(u_i), \phi(v_j)) T_{ij} - \epsilon \sum_{i=1}^{m} \sum_{j=1}^{n} T_{ij} \\ \mathit{s.t.} & T\mathbf{1} \preceq \mathbf{1} \\ & T^\top \mathbf{1} \preceq \mathbf{1} \\ & T_{ij} \in \{0, 1\} \end{array} \tag{1}$$

Definition 2.0.4 (Soft Alignment Problem) *A soft alignment between G_1 and G_2, is a set $\mathcal{M}$ where*

$$\mathcal{M} \subseteq (V_1 \cup \emptyset) \times (V_2 \cup \emptyset)$$

, such that

$$\begin{array}{ll} \sum w_{u_i,\cdot} \leq p_i & \forall u_i \in V_1 \\ \sum w_{\cdot,v_j} \leq q_j & \forall v_j \in V_2 \\ w_{\cdot,\cdot} \geq 0 & \end{array}$$

, where $p \in \Delta_m$ and $q \in \Delta_n$.

Compared to Definition 2.0.2, this definition relaxes the binary constraint for the weight of each alignment. However, we normalize the weights incident on each vertex. For this study, we work with the simple case where $p_i = \frac{1}{m}$ for all $i \in [m]$ and $q_j = \frac{1}{n}$ for all $j \in [n]$ for all $j \in [n]$, but the results can be extended easily to the more general case. The above problem can be formulated as the following linear program:

Definition 2.0.5 (Soft Alignment Optimization Formulation #1)

$$\begin{array}{ll} \mathit{min}_{T \in \mathbb{R}^{m \times n}} & \sum_{i=1}^{m} \sum_{j=1}^{n} d(\phi(u_i), \phi(v_j)) T_{ij} - \epsilon \sum_{i=1}^{m} \sum_{j=1}^{n} T_{ij} \log(T_{ij}) \\ \mathit{s.t.} & T\mathbf{1} \leq p \\ & T^\top \mathbf{1} \leq q \\ & T_{ij} \geq 0 \end{array} \tag{2}$$

2.3 Entropic Regularization

However, solving either the binary integer program 1 or the linear program 2 is hard when m and n are large. In an attempt to balance scalability with accuracy, [11] proposed entropic regularization by adding a negentropy term $E(T)$ to the objective. Let us define the cost matrix $C \in \mathbb{R}^{m \times n}$, with

$$C_{ij} = d(\phi(u_i), \phi(v_j)) - \epsilon$$

Then,

Definition 2.0.6 (Entropic Regularized Soft Alignment)

$$\begin{array}{ll} \mathit{min}_{T \in \mathbb{R}^{m \times n}} & \langle C, T \rangle + \gamma \sum_{i=1}^{m} \sum_{j=1}^{n} E(T_{ij}) \\ \mathit{s.t.} & T\mathbf{1} \leq p \\ & T^\top \mathbf{1} \leq q \\ & T_{ij} \geq 0 \end{array} \tag{3}$$

, where

$$E(x) = \begin{cases} 0 & x = 0 \\ -x \log(x) & x > 0 \end{cases}$$

3 Lagrangian and Dual Analysis for Soft Alignment

In this section, we investigate the dual of the optimization problem observed in Equation (3). First, let us rewrite the optimization problem in (3) with $p = \frac{1}{m}\mathbb{1}$ and $q = \frac{1}{n}\mathbb{1}$ as follows:

$$\begin{aligned} \text{minimize} \quad & \langle t, c\rangle + \gamma \sum_{i=1}^{mn} t_i \log t_i \\ \text{s.t.} \quad & t \succeq 0 \\ & \begin{bmatrix} A_r \\ A_c \end{bmatrix} t \preceq \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \end{aligned} \tag{4}$$

, where $t, c \in \mathbb{R}^{mn}$ are the row-major stacked vectors of the matrices T and C, respectively. The matrices $A_r \in \mathbb{R}^{m \times mn}$ and $A_c \in \mathbb{R}^{n \times mn}$ enforce the row and column constraints, respectively. They are in the form

$$A_r = \left.\begin{bmatrix} \overbrace{1 \;\; 1 \;\; \dots \;\; 1}^{\text{n}} & & & \\ & \overbrace{1 \;\; 1 \;\; \dots \;\; 1}^{\text{n}} & & \\ & & \ddots & \\ & & & \overbrace{1 \;\; 1 \;\; \dots \;\; 1}^{\text{n}} \end{bmatrix}\right\} \text{m rows} \tag{5}$$

$$A_c = \left.\left[\begin{array}{ccc:ccc:c:ccc} 1 & & & 1 & & & & 1 & & \\ & \ddots & & & \ddots & & \dots & & \ddots & \\ & & 1 & & & 1 & & & & 1 \end{array}\right]\right\} \text{n rows} \tag{6}$$

For clarity, let us introduce the notation $A = \begin{bmatrix} A_r \\ A_c \end{bmatrix}$. Then, we introduce Lagrange multipliers $\lambda \in \mathbb{R}^{m+n}$, corresponding to each row of the constraint matrix. Since the constraints are inequalities in this formulation, we have $\lambda \succeq 0$. Then, the Lagrangian is:

$$\mathcal{L}(t, \lambda) = \langle t, c\rangle + \gamma \sum_{i=1}^{mn} t_i \log t_i + \langle \lambda, At - \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}\rangle \tag{7}$$

Isolating the terms dependent on the primal variables t, we get

$$\mathcal{L}(t, \lambda) = \left(\langle t, c + A^\top \lambda\rangle + \gamma \sum_{i=1}^{mn} t_i \log t_i\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}\rangle \tag{8}$$

To obtain the dual, we solve

$$g(\lambda) = \min_{t \succeq 0, \sum t_i = 1} \mathcal{L}(t, \lambda) = \min_{t \succeq 0, \sum t_i = 1} \left(\langle t, c + A^\top \lambda\rangle + \gamma \sum_{i=1}^{mn} t_i \log t_i\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}\rangle \tag{9}$$

Crucially, the function we are minimizing over is separable, i.e.,

$$\langle t, c + A^\top \lambda\rangle + \gamma \sum_{i=1}^{mn} t_i \log t_i = \sum_{i=1}^{mn} \big(t_i(c_i + a_i^\top \lambda) + \gamma t_i \log t_i\big) \tag{10}$$

, where $c_i, a_i \in \mathbb{R}^{m+n}$ are the i-th column of C and A, respectively. Therefore, we can minimize with respect to each component of t, i.e.,

$$g(\lambda) = \left(\sum_{i=1}^{mn} \min_{t_i \geq 0, \sum t_i = 1} \big(t_i(c_i + a_i^\top \lambda) + \gamma t_i \log t_i\big)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}\rangle \tag{11}$$

Taking derivative with respect to t_i and setting it to zero, we get

$$(c_i + a_i^\top \lambda) + \gamma \log \hat{t_i^*} + \gamma = 0 \tag{12}$$

$$-\frac{1}{\gamma}(c_i + a_i^\top \lambda) - 1 = \log \hat{t_i^*} \tag{13}$$

$$\tag{14}$$

which gives

$$\hat{t_i^*} = \exp(-\frac{1}{\gamma}(c_i + a_i^\top \lambda) - 1) \tag{15}$$

Normalizing, i.e., projecting back into Σ_{m+n},

$$t_i^* = \frac{\exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda) - 1)}{\displaystyle\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda) - 1)} = \frac{-\exp(\frac{1}{\gamma}(c_i' + a_i^\top \lambda))}{\displaystyle\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))} \tag{16}$$

Let us denote the normalizing factor Z as

$$Z = \sum_{i=1}^{mn} \exp(\frac{1}{\gamma}(c_i' + a_i^\top \lambda)) \tag{17}$$

Therefore, the dual function is

$$\begin{align}
g(\lambda) &= \left(\sum_{i=1}^{mn} \left(t_i^*(c_i + a_i^\top \lambda) + \gamma t_i^* \log t_i^*\right)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{18} \\
&= \left(\sum_{i=1}^{mn} t_i^*(c_i + a_i^\top \lambda + \gamma \log t_i^*)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{19} \\
&= \left(\sum_{i=1}^{mn} \frac{\exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))}{Z}(c_i + a_i^\top \lambda + \gamma \log \left(\frac{\exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))}{Z}\right))\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{20} \\
&= \frac{1}{Z}\left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))(c_i + a_i^\top \lambda - \gamma(\frac{1}{\gamma}(c_i' + a_i^\top \lambda)) - \gamma \log Z)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{21} \\
&= \frac{1}{Z}\left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))(c_i + a_i^\top \lambda - (c_i' + a_i^\top \lambda) - \gamma \log Z)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{22} \\
&= -\frac{\gamma}{Z}\left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))(\log Z)\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{23} \\
&= -\frac{\gamma}{Z}\log Z\left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{24} \\
&= -\gamma \log Z - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{25} \\
&= -\gamma \log \left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i + a_i^\top \lambda))\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \tag{26}
\end{align}$$

Now we are interested in solving the dual optimization problem:

$$\max_{\lambda \succeq 0} \left\{ g(\lambda) = -\gamma \log \left(\sum_{i=1}^{mn} \exp(-\frac{1}{\gamma}(c_i + a_i^\top \lambda))\right) - \langle \lambda, \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \rangle \right\} \tag{27}$$

Which has a closed-form gradient

$$\begin{align}
[\nabla g(\lambda)]_i &= -\gamma \left(\frac{-\frac{1}{\gamma}\exp(-\frac{1}{\gamma}(c_i' + a_i^\top \lambda))a_i}{Z}\right) - [\begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}]_i \tag{28} \\
&= t_i^* a_i - [\begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix}]_i \tag{29}
\end{align}$$

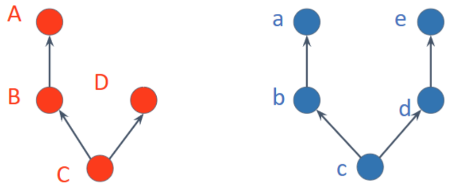

Figure 1: The two simple graphs used as toy examples.

In matrix form,

$$\nabla g(\lambda) = At^*(\lambda) - \begin{bmatrix} \frac{1}{m}\mathbb{1}_m \\ \frac{1}{n}\mathbb{1}_n \end{bmatrix} \tag{30}$$

and closed-form hessian

$$\nabla^2 g(\lambda) = \frac{1}{\gamma} A \left(\text{diag}(t^*) - t^* {t^*}^\top \right) A^\top \tag{31}$$

, where

$$t^*(\lambda) = \frac{1}{Z} \begin{bmatrix} \exp(\frac{1}{\gamma}(c'_1 + {a_1}^\top \lambda) - 1) \\ \exp(\frac{1}{\gamma}(c'_2 + {a_2}^\top \lambda) - 1) \\ \vdots \\ \exp(\frac{1}{\gamma}(c'_{mn} + {a_{mn}}^\top \lambda) - 1) \end{bmatrix} \tag{32}$$

represents the primal and dual variable correspondence.

4 Experiments

All code used to generate the following experiments can be found and replicated in the GitHub link(Maybe not yet public).

4.1 Alignment between toy graphs

In this section, we present a toy example of alignment between two simple graphs with $m = 4$ and $n = 5(1)$. The goal of this section is twofold:

1. To demonstrate the feasibility of the two modes of alignments defined above, and
2. to perform elementary side-by-side analysis between hard and soft alignments.

The four optimal hard alignments for this toy example are depicted on the left side of Figure 3.

For obtaining the soft alignments, vanilla gradient ascent is used[1] to solve Problem 27. A step size of $\eta = 0.01$, a nonalignment penalty of $\epsilon = 0.1$, and a regularization constraint of $\gamma = 0.1$ were used. The resulting soft alignment is plotted in Figure 2, and the alignment matrix is depicted in Figure 3.

For this toy example, we make an interesting observation: the soft alignment behaves almost like a normalized convex combination of the hard alignments. To see this, observe that in all four optimal hard alignments, the node c from the red graph is aligned to the node C from the blue graph. Therefore, in any convex combination of them, the $(3, 3)$ entry will always be one. Correspondingly, in the optimal soft alignment, the (c, C) alignment is given is a weight of 0.2, the upper bound for a single entry in the soft alignment matrix ($\min(\frac{1}{m}, \frac{1}{n})$). Moreover, among the four optimal alignments, node A from the red graph is aligned to either a or e from the blue graph. This is reflected in the soft alignment matrix, where the $(1, 1)$ and $(1, 5)$ entries are both 0.125, nearly at the halfway point between the two extremes of 0 and 0.2. Similar arguments can be made for most other entries in the soft alignment matrix, execpt for the nonzero $(3, 1), (3, 5), (4, 1), (4, 5)$ entries, where none of the optimal hard alignments align the corresponding nodes. The author hypothesize that this is

[1] Work in progress. currently implementing: momentum accelerated methods[4], (tentative) second-order methods, and optimization with sparsity constraints[8].

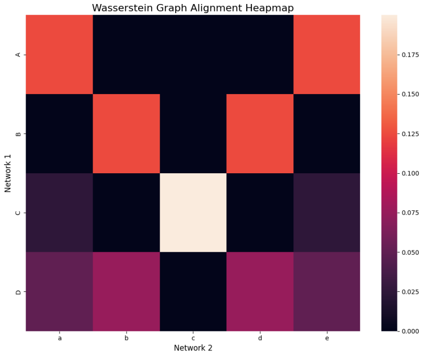

Figure 2: Soft alignment heatmap between toy example graphs

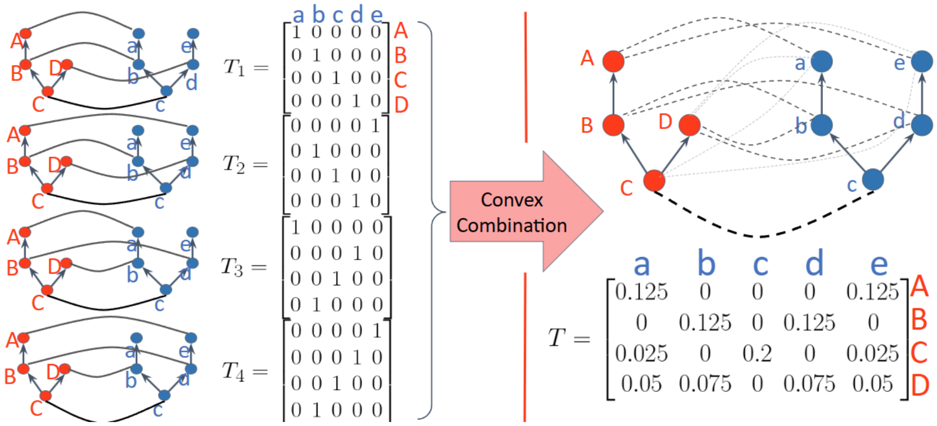

Figure 3: Side-by-side comparison of hard and soft alignment on toy example. It can be hypothesized that the soft alignment(right) can be formulated as a convex combination (plus some error) of the hard alignments(left)

phenomenon could be attributed to the curvature of the feasibility region by the introduction of entropic regularization, and depends on γ.

It remains unclear whether this phenomenon is a coincidence or a general property of the soft alignment problem, as it becomes increasingly difficult to visualize the soft alignment matrix as the size of the graphs increase. In the next section, we will explore this phenomenon in a more complex setting, by looking into both alignments between real-world graphs.

4.2 Alignment between Sub-Saharan African food webs

In this section, we present a specific instance of the graph alignment problem in computational Ecology. In particular, we are interested in aligning food webs from different ecosystems to identify common structural roles that species play in their respective food webs. For example, if species A from food web α is aligned to species B from food web β, we expect that species A plays a very similar structural role in the context of food web α as compared to species B in food web β.

Previous approaches utilize the simulated annealing approach to solve the hard alignment problem 1, and reasons about the structural backbones of the food webs based on the solution[1]. However, due to the nature of the hard alignment problem, this approach is not only computationally expensive, but also only captures a single optimal hard alignment. This leads to a myopic view of the structures that each species plays in the food web. In this work, we will explore the soft alignment problem 2 to identify the structural backbones of the food webs.

In this context, the feature map ϕ is defined as the mapping from the species in the food webs to the vector of counts of the distinct roles of network motifs that each species plays in the food web[14].

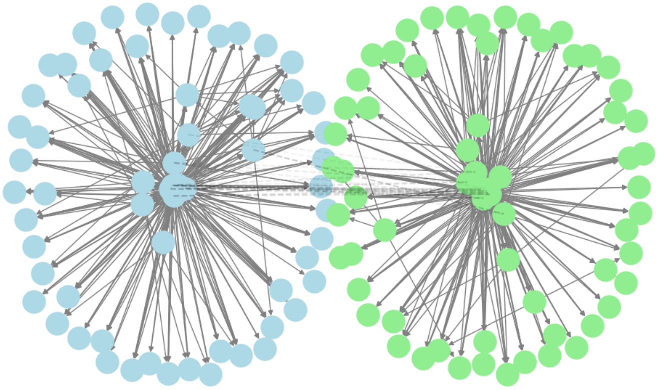

Figure 4: A visualization of the two food webs used in this experiment. The left graph is the Cliff of Bandiagara food web, and the right graph is the Namaqua food web. Each node represents a species, and the edges represent the (predator → prey) interactions between species.

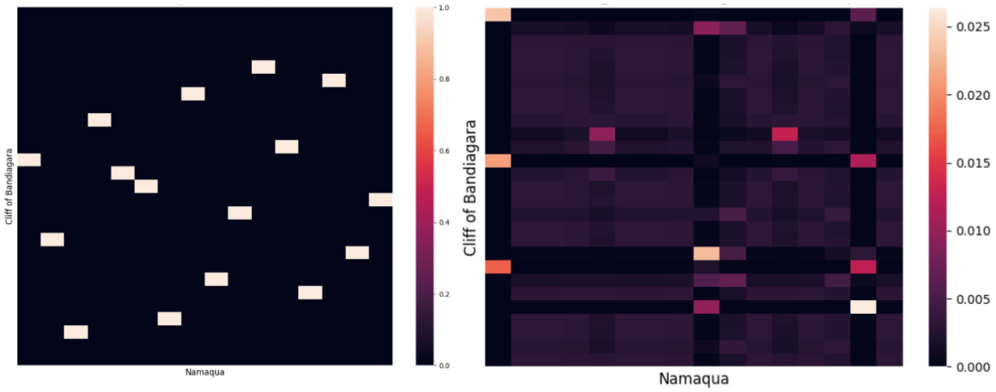

Figure 5: Side-by-side comparison of the hard alignment(left) and soft alignment(right) between the Cliff of Bandiagara and Namaqua food webs.

The metric $d(\cdot, \cdot)$ in the feature space is defined as $d(u_i, v_j) = 1 - \rho(u_i, v_j)$ for species u_i and v_j, where $\rho(\cdot, \cdot)$ is the Pearson correlation coefficient.

Two relatively small food webs from the sub-Saharan African region are selected for this experiment. The first food web is the *Cliff of bandiagara* food web, which contains $m = 27$ species, and the second food web is the *Namaqua* food web, which contains $n = 16$ species. The two food webs are depicted in Figure 4.

Note that is it no longer computationally tractable to compute the full set of optimal hard alignments for these two food webs. Therefore, we will only compute a single hard alignment between them, as observed in [1]. The resulting optimal hard alignment is depicted on the left in Figure 5. Moreover, we compute the soft alignment between the two food webs using vanilla gradient ascent with $\eta = 0.1, \epsilon = 0.2, \gamma = 0.1$. The resulting soft alignment is depicted on the right in Figure 5.

In general, the two alignments seem to largely agree on the alignment between the predators of the food webs, with high out-degrees, and do not agree on the alignment between prey, which are often leaf nodes. This general behavior is expected, because the local structural information of the leaf nodes is exactly the same, which makes the algorithm treat them like symmetric (or “identical") nodes. For the predators, they have large counts of motifs where a node consumes the other two, which results in them being clustered in the feature space, away from the rest of the nodes. Therefore, both hard alignments and soft alignments can associate them together. However, in the case of the prey nodes, the hard alignment is forced to pair them up, until all the preys in one of the food webs are exhausted, and the rest of the preys are forced to align to other species, which could potentially generate large costs and inaccurate structural similarity matchings. However, for soft alignments, the algorithm is able to “split" the alignment for a single species, causing each of the prey nodes of the first graph to be equally aligned to each of the prey nodes of the second graph, even if the difference between the sizes of the graphs is large.

Furthermore, soft alignments give more fine-grained insights into the roles that the predators play. To illustrate this point, let us focus on the first column of both heatmaps in Figure 5: it represents the alignment distribution of the species `caracal caracal`, or the caracal (as the name might suggest), a predator in the food web located at *Namaqua*. In the hard alignment, the caracal is aligned also to the caracal living in *Cliff of Bandiagara*, which is ecologically sensible: the same species will likely exhibit similar behavior and hence serve similar roles in different food webs. However, the soft alignment provides perhaps a more complete perspective: the caracal in *Namaqua* is aligned strongly to the following three species in *Cliff of Bandiagara*:

- `panthera pardus`, the leopard,
- `caracal caracal`, the caracal, and
- `crocuta crocuta`, the spotted hyena.

This tells us not only that the caracal in *Namaqua* serves a similar role to its *Cliff of Bandiagara* counterpart, but also that there are two more species in *Cliff of Bandiagara* that serve a very similar structural role to the caracal in *Namaqua*. This indeed matches with traditional ecological understanding[2]. Moreover, we are able to quantify exactly *how much* similar these structural roles are, relative to each other, using the weights associated with the alignments.

5 Conclusion and Future Work

In this report, we explored the formulation of hard and soft alignment problems between graph. We introduced a way to make the soft alignment problem easier to solve by adding an entropic regularization term, so that we can perform primal-dual analysis on the problem and turn it into a smooth unconstrained optimization problem with closed form gradients and hessians. This let us use gradient-based methods to find good solutions even for larger graphs.

Our experiments on small toy graphs and real-world food webs showed that soft alignments can often behave like a blend of several hard alignments. This gives us more flexibility and insight, especially when the graphs are large or when there are multiple reasonable alignments.

Overall, soft alignment methods can be a useful alternative to hard alignments, especially when we want to understand deeper structural patterns or when a single matching doesn’t tell the full story.

For the future, we are exploring three facets of improvement based on the current results:

1. Firstly, we are looking into implementing accelerated methods, as well as second order methods(since we have a closed form hessian) to perform optimization for the soft alignment problem. We believe that it will speed up the optimization even further, making this more feasible and scalable, and hopefully more robust to hyperparameters.
2. Secondly, we are exploring a change of approach to designing the cost function: instead of minimizing the sum of the costs between aligned species, we are exploring the minimization of the cost between the neighbors of the species we wish to align. We call this first-order alignment, and it turns out that this gives us a Gromov-Wasserstein-like problem.
3. Thirdly, we would like to dive deeper into interpreting and analyzing the difference between hard alignments and soft alignments. Personally, I am interested in exploring the geometry of the feasible sets of the two problems, as well as how the solutions of the problems compare geometrically.

[2] For more information, please consult the outstanding biologists who are working with me.

References

[1] Bernat Bramon Mora, Dominique Gravel, Luis J Gilarranz, Timothée Poisot, and Daniel B Stouffer. Identifying a common backbone of interactions underlying food webs from different ecosystems. *Nature Communications*, 9(1):2603, 2018.

[2] Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. Cross-network embedding for multi-network alignment. In *The world wide web conference*, pages 273–284, 2019.

[3] Jingyuan Duan, Zhao Kang, Ling Tian, and Yichen Xin. Multi-level social network alignment via adversarial learning and graphlet modeling. *Neural Networks*, page 107230, 2025.

[4] Pavel Dvurechensky, Petr Ostroukhov, Alexander Gasnikov, César A Uribe, and Anastasiya Ivanova. Near-optimal tensor methods for minimizing the gradient norm of convex functions and accelerated primal–dual tensor methods. *Optimization Methods and Software*, 39(5):1068–1103, 2024.

[5] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[6] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL `https://www.gurobi.com`.

[7] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[8] Tianlin Liu, Joan Puigcerver, and Mathieu Blondel. Sparsity-constrained optimal transport. *arXiv preprint arXiv:2209.15466*, 2022.

[9] Ali Masoudi-Nejad, Falk Schreiber, and Zahra Razaghi Moghadam Kashani. Building blocks of biological networks: a review on major network motif discovery algorithms. *IET systems biology*, 6(5):164–174, 2012.

[10] Rob Patro and Carl Kingsford. Global network alignment using multiscale spectral signatures. *Bioinformatics*, 28(23):3105–3114, 2012.

[11] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[12] Shruti Saxena and Joydeep Chandra. A survey on network alignment: approaches, applications and future directions. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 8216–8224, 2024.

[13] Ziying Song, Haiyue Wei, Lin Bai, Lei Yang, and Caiyan Jia. Graphalign: Enhancing accurate feature alignment by graph matching for multi-modal 3d object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3358–3369, 2023.

[14] Julia Tavella, Fredric M Windsor, Débora C Rother, Darren M Evans, Paulo R Guimaraes Jr, Tania P Palacios, Marcelo Lois, and Mariano Devoto. Using motifs in ecological networks to identify the role of plants in crop margins for multiple agriculture functions. *Agriculture, Ecosystems & Environment*, 331:107912, 2022.