SAND2015-9891R

# Triangular Alignment (TAME): A Tensor-based Approach for Higher-order Network Alignment

Shahin Mohammadi, David F. Gleich, Tamara G. Kolda, and Ananth Grama

**Abstract**—Network alignment is an important tool with extensive applications in comparative interactomics. Traditional approaches aim to simultaneously maximize the number of conserved edges and the underlying similarity of aligned entities. We propose a novel formulation of the network alignment problem that extends topological similarity to higher-order structures and provide a new objective function that maximizes the number of aligned substructures. This objective function corresponds to an integer programming problem, which is NP-hard. Consequently, we approximate this objective function as a surrogate function whose maximization results in a tensor eigenvalue problem. Based on this formulation, we present an algorithm called Triangular AlignMEnt (TAME), which attempts to maximize the number of aligned triangles across networks. We focus on alignment of triangles because of their enrichment in complex networks; however, our formulation and resulting algorithms can be applied to general motifs. Using a case study on the NAPABench dataset, we show that TAME is capable of producing alignments with up to 99% accuracy in terms of aligned nodes. We further evaluate our method by aligning yeast and human interactomes. Our results indicate that TAME outperforms the state-of-art alignment methods both in terms of biological and topological quality of the alignments.

**Index Terms**—Graphs and networks, Optimization, Higher-order network alignment, Tensor Z-eigenpair, SS-HOPM

## 1 Introduction

Modeling cellular machinery as a network of interacting biomolecules provides significant opportunities for understanding and controlling various biological processes. This complex network, or *interactome*, may include direct relationships among biomolecules, such as physical, regulatory, or signaling interactions, or indirect phenotypic relationships such as epistatic interactions. One common abstraction is a protein-protein interaction network (PPI), which is an undirected graph where nodes represent proteins and edges encode physical interactions among pairs of proteins. Protein-protein interaction networks are extensively used for modeling and understanding pathways and protein complexes with respect to their organization and function.

Network motifs are one means of identifying organization and function in these networks. A network motif is a connected subgraph that occurs with significantly higher frequency compared to an ensemble of random graphs with the same size and degree distribution. Over-representation of these patterns are hypothesized to be related to their functional significance [1]; and, indeed, network motif analysis uncovers the fundamental circuits that are repeatedly used to perform critical functions within the cell. Due to their important role in decoding biological networks, various algorithms have been proposed in literature for network motif detection [2]–[6]. These methods have identified key motifs, such as feed forward and feedback cycles in directed networks and triangles in undirected graphs. Furthermore, these motifs are shown to be involved in regulating cell function, as well as influencing global network characteristics [7]–[11].

Concurrent with the development of methods for network motif detection, there has been ongoing work on *network alignment* algorithms for identification of conserved modules across networks. The goal of network alignment is to identify a mapping between nodes of networks that maximizes similarity (as defined by a suitable measure) between mapped entities. These mappings can be used to infer orthologies for unannotated proteins, as well as transferring known biology regarding common pathways, protein complexes, recurring building blocks, and missing interactions. These conserved substructures are important since cellular functions require all of their constituent components (nodes and their interactions) to be conserved. Conversely, conserved modules provide insights into corresponding functional organization [12].

The network alignment problem, unlike its counterpart over sequences, is NP-complete to solve exactly, since in the most generic form in can be reduced to the subgraph isomorphism problem. However, different heuristics have been proposed to address this complexity through suitable reformulation, as well as incorporating additional data to guide the alignment process. We survey these methods in Section 2.3 in more detail. An important distinction among alignment methods is their local versus global nature. Local alignment methods aim to identify conserved functional modules between networks, such as signaling pathways and

- *S. Mohammadi, D. F. Gleich, and A. Grama are with the Department of Computer Science, Purdue University, West Lafayette, IN 47907 E-mail: {mohammadi, dgleich, ayg}@purdue.edu.*
- *T. G. Kolda is with the Sandia National Laboratories, Livermore, CA 94551. E-mail: tgkolda@sandia.gov.*

*Manuscript received April 19, 2005; revised September 17, 2014.*

protein complexes, by optimally aligning these substructures. Due to duplication-divergence events, there can be more than one match for each substructure, which makes alignments ambiguous. Global network alignment, on the other hand, aims to identify a one-to-one mapping between all pair of nodes in the input graphs that maximizes similarity of aligned nodes.

In this paper, we introduce a new class of methods based on *higher-order network alignment*. These methods combine strengths of both global and local network alignment. In this framework, users can define any network motif structure of interest to drive the alignment process. These general structures can be represented through a *motif tensor*, in which the order of tensor is the same as the size of the given subgraph template. We encode the higher order network using a tensor-based formulation and show that the exact solution to the alignment problem is equivalent to solving a higher-order integer program, which is NP-hard. To optimize this NP-hard objective on large networks, we exploit a bijection between the eigenpairs of the motif tensor and a heuristic approximation of the integer program. We can then use the previously proposed SS-HOPM [13] method to identify maximizing dominant eigenpairs of a symmetric tensor, and propose a higher-order alignment method based on this scheme. The motif tensor of the alignment graph can be represented as the Kronecker product of motif tensors for each input graph. This tensor is too large to fit in the memory for typical PPI networks, even for small motifs. We develop a novel implicit kernel for computing the tensor-vector product as the main operator within SS-HOPM. Similar kernels have been previously proposed in the context of computer vision research [14], [15]; however, we present a highly efficient, motif-centered version that is easily extendible to higher order sub-structures.

Using a case study of triangle motifs, we present a complete algorithm, called *Triangular AlignMEnt (TAME)*. Furthermore, we propose a constrained variant of our algorithm, *cTAME*, that operates only on a subset of reliable nodes. This method provides better accuracy in cases where a significant fraction of triangles in input graphs are false-positives. We compare our method to the state-of-art methods for network alignment and illustrate its superior performance on both synthetic datasets from NAPAbench [16] and alignment of yeast and human interactomes. Our framework can be easily extended to arbitrary subgraphs, and suggests an alternative view to the network alignment problem.

## 2 Notations and Terminology

### 2.1 Graphs and Hypergraphs

Biochemical networks are often modeled as graphs in which vertices (or nodes) represent biomolecules (proteins, genes, etc.) and edges (or arcs) encode pairwise relationships among them. Formally, a graph $\boldsymbol{G}$ is represented by $\boldsymbol{G} = (V_G, E_G)$, where $V_G$ is a finite set of vertices, $V_G = \{v_1, v_2, \ldots, v_n\}$, and $E_G$ is a finite set of edges, denoted by $(v_i, v_j)$, such that $E_G \subseteq (V_G \times V_G)$. We focus on undirected graphs, where edges define a symmetric relation among graph vertices. A graph can be represented by a matrix $\mathbf{A}_{\boldsymbol{G}}$ of size $|V_G| \times |V_G|$, known as the *adjacency matrix*, in which $\mathbf{A}_G(i,j) = 1$ iff $(v_i, v_j) \in E_G$. The graph neighborhood for each node $v_i$ in the graph, represented by $N_G(i)$, is defined as the set of nodes that have an edge with $v_i$; formally $N_G(i) = \{j \mid (i,j) \in E_G\}$. Given a pair of graphs, $\boldsymbol{G} = (V_G, E_G)$ and $\boldsymbol{H} = (V_H, E_H)$, their Kronecker product is a graph with $|V_G| \times |V_H|$ vertices in which there is an edge between nodes $ii'$ and $jj'$ iff $(i,j) \in E_G$ and $(i',j') \in E_H$.

Hypergraphs are natural generalizations of graphs in which the relations among vertices is not restricted to be pairwise. Formally, a hypergraph is defined using the pair $\boldsymbol{\mathcal{G}} = (V_G, \mathcal{E}_G)$, where $V$ is the set of vertices and $\mathcal{E}$ is the set of *hyperedges*. Here, each hyperedge defines a relationship among a nonempty subset of vertices. A *$k$-uniform* hypergraph is a hypergraph in which the cardinality of each hyperedge is exactly $k$. As such, 2-uniform hypergraphs are equivalent to traditional graphs. A $k$-uniform hypergraph can be represented by a $k^{th}$-order tensor $\mathcal{T}_G$, known as the *adjacency tensor*, where $\mathcal{T}_G(i_1, i_2, \ldots, i_k) = 1$ iff $(i_1, i_2, \ldots, i_k) \in \mathcal{E}_G$. The hypergraph incidence set for each node $v_i$ in a $k$-uniform hypergraph, represented by $\mathcal{N}_G(i)$, is the set of $(k-1)$-node subsets such that adding node $i$ to each subset forms an edge. This is one possible generalization of a neighborhood to a hypergraph, and is formally defined as $\mathcal{N}_G(i) = \{(i_2, i_3, \ldots, i_k) \mid (i, i_2, i_3, \ldots, i_k) \in \mathcal{E}_G\}$. For a given graph $\boldsymbol{G} = (V_G, E_G)$ and a given size $k$ structural motif $\boldsymbol{M} = (V_M, E_M)$, where $|V_M| = k$, we can represent the occurrences of $\boldsymbol{M}$ in $\boldsymbol{G}$ by a $k$-way tensor $\mathcal{T}_G$, referred to as the *motif-tensor*, where $\mathcal{T}_G(i_1, \ldots, i_k) = 1$ iff the induced subgraph among vertices $\{v_{i_1}, \ldots, v_{i_k}\}$ in $\boldsymbol{G}$ is isomorphic to $\boldsymbol{M}$. Throughout this paper, we make extensive use of a special case of the motif-tensor where the substructure of interest is the triangle motif. Given an undirected graph $\boldsymbol{G}$, its *triangle tensor*, denoted by $\triangle_G$, is a third-order tensor such that $\triangle_G(i,j,k) = 1$ iff $(v_i, v_j), (v_j, v_k), (v_k, v_i) \in E_G$.

### 2.2 Tensor definition and properties

A real-valued $m^{th}$-order $n$-dimensional tensor, denoted by $\mathcal{T}^{[m,n]}$, is a multiway array, whose entries can be indexed using an $m$-dimensional tuple, and each tensor way (or mode) has dimension $n$. An $n$-dimensional vector and square matrix are examples of $1^{st}$-order and $2^{nd}$-order tensors, respectively. We refer to elements of a tensor $\mathcal{T}^{[m,n]}$ using $\mathcal{T}(i_1, i_2, \ldots, i_m)$ and $\mathcal{T}_{i_1, i_2, \ldots, i_m}$, interchangeably. A tensor $\mathcal{T}$ is *symmetric* iff:

$$\mathcal{T}(i_1, i_2, \ldots, i_m) = \mathcal{T}(i_{\pi(1)}, i_{\pi(2)}, \ldots, i_{\pi(m)}) \tag{1}$$

for all $\pi \in \Pi_m$, where $\Pi_m$ is the set of all permutations of $(1, 2, \ldots, m)$. As an example, note that the triangle tensor is a symmetric tensor.

Given a symmetric tensor $\mathcal{T}^{[m,n]}$, together with an $n$-dimensional vector $\boldsymbol{x} \in \mathbb{R}^n$, we concern ourselves with two operations. The first is the tensor-vector product: $\mathcal{T}\boldsymbol{x}^{m-1}$, which is defined element-wise as

$$(\mathcal{T}\boldsymbol{x}^{m-1})_{i_1} = \sum_{i_2=1}^{n} \sum_{i_3=1}^{n} \cdots \sum_{i_m=1}^{n} \mathcal{T}_{i_1, i_2, \ldots, i_m} x_{i_2} x_{i_3} \ldots x_{i_m}. \tag{2}$$

The second is the scalar polynomial form in $\boldsymbol{x}$: $\mathcal{T}\boldsymbol{x}^m$ defined as

$$\mathcal{T}\boldsymbol{x}^m = \sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\sum_{i_3=1}^{n}\cdots\sum_{i_m=1}^{n}\mathcal{T}_{i_1,i_2,\ldots,i_m}x_{i_1}x_{i_2}x_{i_3}\ldots x_{i_m}. \tag{3}$$

Note that $\boldsymbol{x}^T(\mathcal{T}\boldsymbol{x}^{m-1}) = \mathcal{T}\boldsymbol{x}^m$.

A pair $(\lambda, \boldsymbol{x})$, $\lambda \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^n$, is the Z-eigenpair of the symmetric tensor $\mathcal{T}$, iff:

$$\mathcal{T}\boldsymbol{x}^{m-1} = \lambda\boldsymbol{x};\ \text{with}\ \|\boldsymbol{x}\|_2 = 1. \tag{4}$$

Any eigenpair $(\lambda, \boldsymbol{x})$ of tensor $\mathcal{T}$ is a Karush-Kuhn-Tucker (KKT) point of the following nonlinear optimization problem [17]:

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{maximize}} \quad & \mathcal{T}\boldsymbol{x}^m \\ \text{subject to} \quad & \|\boldsymbol{x}\|_2 = 1. \end{aligned} \tag{5}$$

We will use one additional concept: the Kronecker product of tensors. This type of product is a reshaped version of the well-established outer product for tensors. Formally, given a pair of symmetric tensors, $\mathcal{T}_1 \in \mathcal{R}^{[m,n_1]}, \mathcal{T}_2 \in \mathcal{R}^{[m,n_2]}$, their Kronecker product, denoted by $\mathcal{T}_1 \otimes \mathcal{T}_2 \in \mathcal{R}^{[m,n_1n_2]}$, is defined as:

$$(\mathcal{T}_1 \otimes \mathcal{T}_2)(i_1i'_1,\ldots,i_mi'_m) = \mathcal{T}_1(i_1,\ldots,i_m)\mathcal{T}_2(i'_1,\ldots,i'_m) \tag{6}$$

with $1 \le i_1,\ldots,i_m \le n_1$, $1 \le i'_1,\ldots,i'_m \le n_2$, and the notation $i_ki'_k$ denotes a specific index for the index representing the pair which is $(i_k-1)n_2 + i'_k$.

### 2.3 Overview of network alignment methods

Based on the alignment strategy, we can generally classify different methods as either *local* or *global* alignment techniques. Local alignment aims to identify common substructures corresponding to pathways or protein complexes that are conserved in networks of different species. These alignments are consistent with duplication/divergence model of evolution. However, they typically yield ambiguous mappings since functional building blocks can have many-to-many relationships. On the other hand, global alignment attempts to find the best overall mapping between the nodes of input graphs that maximizes both functional and topological similarity of aligned nodes, while enforcing the one-to-one constraint. In pairwise alignment, this leads to a unique alignment for each node in the smaller graph to a maximum of one node in the larger graph. These alignments are unambiguous and can be used to transfer functional orthologies between pairs of proteins, as well as to compute the overall similarity of input graphs.

Local alignment methods generally involve a scoring function to evaluate a local alignment, and a search method to identify high scoring alignments. PathBlast [19], [20] and NetworkBlast [21], [22] are early examples that used a probabilistic scoring function. Flannick *et al.* [23] proposed an evolutionarily-motivated scoring function and incorporated known alignments via a supervised learning scheme [24]. Koyuturk *et al.* [25], [26] instead poses the local alignment as a suitably formulated optimization problem in their MaWish framework. All of these methods can be tuned to seek local alignments of particular motifs by customizing their scoring and search functions. However, they all suffer the amiguity that results from local alignment.

In the class of global network alignment techniques, there are two prominent paradigms. The first is exemplified by the early method IsoRank [27], [28]. The main idea behind IsoRank is that a pair of nodes corresponds to a good match if the nodes are homologous and their respective neighborhoods are similar. This recursive scheme is then cast as an eigenvalue problem, the solution of which is identified using power method. More generally, these methods utilize heuristic approximations of the alignment on the joint topology of the product graph and relationships with optimization problems, as proposed by Klau *et al.* [29], [30] and Bayati *et al.* [31]. Bayati et al. include optimizations that restrict the search space to the subset of promising candidates for which there is an evidence of sequence similarity. The second class is exemplified via GRAAL [32]–[35] and is based on the notion of computing distances between local signatures to indicate topological similarity. Patro *et al.* [36] proposed a novel topological signature that is based on the spectrum of the normalized Laplacian for the induced subgraph around each node. Many of these methods include specific techniques to turn the topological information into a one-to-one alignment.

The methods proposed in this paper follow the global alignment paradigm more closely but incorporate some aspects of the local alignment ideas through the specification of the alignment motif.

## 3 Higher-Order Network Alignment

Our framework for higher-order network alignment draws heavily on the integer quadratic program for global network alignment. We begin by reviewing this formulation [29]–[31].

### 3.1 Formulation of global network alignment as a Binary Quadratic Program (BQP)

Given a pair of networks, represented by $\boldsymbol{G} = (V_G, E_G)$ and $\boldsymbol{H} = (V_H, E_H)$, the global network alignment problem aims to find an optimal one-to-one mapping between vertices of $\boldsymbol{G}$ and $\boldsymbol{H}$ that maximizes both the prior (known) similarity and a topological similarity among pairs of aligned nodes. The topological similarity is the number of *edges* preserved in both $\boldsymbol{G}$ and $\boldsymbol{H}$ under the matching.

Let $\mathbf{W}$ represent the known similarity between graph $\boldsymbol{G}$ and $\boldsymbol{H}$ where $\mathbf{W}(i,i')$ is positive if there is some additional belief that node $i$ in $\boldsymbol{G}$ will map to node $i'$ in $\boldsymbol{H}$ and 0 otherwise. (Here, we use notation where $i$ or $i'$ refers both to a row or column, respectively, of the matrix $\mathbf{W}$ as well as a vertex in $\boldsymbol{G}$ or $\boldsymbol{H}$.) In the case of aligning protein-protein interaction networks, $\mathbf{W}$ is often constructed from the sequence similarity of genes or proteins corresponding to each node pair in $\boldsymbol{G}$ and $\boldsymbol{H}$. Other measures to encode known prior similarities have also been proposed, including structural similarity of proteins, co-pathway memberships, and GO semantic similarities. In this paper, we use $\mathbf{W}$ to encode sequence similarities computed using the Smith-Waterman algorithm, as described in Section 4.1.2.

The matching solution variable is a binary indicator matrix $\mathbf{X}$, where $\mathbf{X}(i,i') = 1$ if node $v_i \in \boldsymbol{G}$ is matched

to node $v_{i'} \in \boldsymbol{H}$, and 0 otherwise. Then the global network alignment problem is equivalent to:

$$\begin{aligned}
&\underset{\mathbf{X}\in\mathbb{Z}_2^{|V_G|,|V_H|}}{\text{maximize}} \\
&\quad (1-\alpha)\cdot \underbrace{\sum_{i\in V_G, i'\in V_H} X(i,i')W(i,i')}_{\text{known similarity}} + \\
&\quad \alpha\cdot\frac{1}{2}\underbrace{\sum_{\substack{i\in V_G, i'\in V_H\\ j\in V_G, j'\in V_H}} X(i,i')X(j,j')A_{\boldsymbol{G}}(i,j)A_{\boldsymbol{H}}(i',j')}_{\text{topological similarity}} \\
&\text{subject to} \\
&\quad \textstyle\sum_{i'\in V_H} X(i,i') \le 1 \text{ for all } i \in V_G \\
&\qquad \textit{(each node in } V_G \textit{ is matched to only 1 in } V_H\textit{)} \\
&\quad \textstyle\sum_{i\in V_G} X(i,i') \le 1 \text{ for all } i' \in V_H \\
&\qquad \textit{(each node in } V_H \textit{ is matched to only 1 in } V_G\textit{).}
\end{aligned} \tag{7}$$

Under the contraints of the problem, the topological similarity term $X(i,i')X(j,j')A_{\boldsymbol{G}}(i,j)A_{\boldsymbol{H}}(i',j')$ is 1 if node $i$ is mapped to node $i'$, node $j$ is mapped to node $j'$, and edges $(i,j)$ and $(i',j')$ exist in $\boldsymbol{G}$ and $\boldsymbol{H}$, respectively. Note that because of symmetry in $A_{\boldsymbol{G}}, A_{\boldsymbol{H}}$, this expression double-counts the aligned edge and hence we divide the total count by 2 to get the topological similarity.

It is convenient to introduce some additional notation to simplify the statement of this problem and explain our generalization. We first define a canonical labeling on the pairs of vertices from these graphs:

$$\ell : (V_G, V_H) \mapsto \text{ a unique integer index in } \{1\cdots|V_G|\cdot|V_H|\}, \tag{8}$$

that is, $\ell(i,i')$ returns a unique index that represents the pair of vertices $i \in V_G$ and $i' \in V_H$. We typically use $\ell(i,i') = (i'-1)|V_G| + i$, a columnwise vectorization of $\mathbf{X}$. This labeling is used to turn the matrices $\mathbf{X}$ and $\mathbf{W}$ into *vectors*. Let $\boldsymbol{x} = \mathbf{vec}(\mathbf{X})$ be a representation of the matrix $\mathbf{X}$ as a length $|V_G||V_H|$ vector where $x(\ell(i,i')) = X(i,i')$. When clear from context, we use the shorthand:

$$x_{\ell(i,i')} \leftrightarrow x_{ii'}. \tag{9}$$

For instance, the known similarity objective then becomes:

$$\sum_{i\in V_G, i'\in V_H} X(i,i')W(i,i') \leftrightarrow \sum_{ii'} x_{ii'}w_{ii'}. \tag{10}$$

To represent the matching constraint on $\boldsymbol{x}$, we define matrix $\mathbf{C} \in \{0,1\}^{(|V_G|+|V_H|)\times|V_G||V_H|}$, where each row represents a vertex in either $\boldsymbol{G}$ or $\boldsymbol{H}$ and columns encode potential pairs of vertices between the two graphs. Formally we can write this as:

$$\begin{aligned}
&\mathbf{C} = \begin{bmatrix} \mathbf{C}_G \\ \mathbf{C}_H \end{bmatrix} \text{ where} \\
&\quad \mathbf{C}_G(i,\ell(i,i')) = 1 \text{ for all } i \in V_G, i' \in V_H \\
&\quad \mathbf{C}_H(i',\ell(i,i')) = 1 \text{ for all } i \in V_G, i' \in V_H
\end{aligned} \tag{11}$$

where $\mathbf{C}_G, \mathbf{C}_H$ are zero everywhere else. Put yet another way, $\mathbf{C}$ is the node-edge incidence matrix of the complete bipartite graph on $|V_G|$ and $|V_H|$ vertices where the edge order is given by the labeling $\ell$. Using the matrix $\mathbf{C}$, the two sets of matching constraints are written: $\mathbf{C}\boldsymbol{x} \le \mathbf{1}_{|V_G|+|V_H|}$, where $\mathbf{1}_{|V_G|+|V_H|}$ is a vector of all ones of length $|V_G|+|V_H|$.

Finally, we introduce one additional binary matrix $\mathbf{S}$ of size $|V_G||V_H| \times |V_G||V_H|$ to describe the set of potentially aligned edges in order to compute the topological similarity. We index the rows and columns of this symmetric matrix using a *pair* of vertices from graph $\boldsymbol{G}$ and $\boldsymbol{H}$; and, hence, our labeling function $\ell$. Let $\mathbf{S}(ii',jj') = 1$, if $(i,j) \in E_G$ and $(i',j') \in E_H$, and 0, otherwise. Using this notation, the total number of conserved edges under alignment represented by $\boldsymbol{x}$ can be computed as $\frac{1}{2}\boldsymbol{x}^T\mathbf{S}\boldsymbol{x}$. Note that when $\ell$ results from a columnwise vectorization of the matrix $\mathbf{X}$ then $\mathbf{S} = A_{\boldsymbol{H}} \otimes A_{\boldsymbol{G}}$ for the Kronecker product.

Using this new notation, we can write the global network alignment as the following binary quadratic program (BQP):

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{maximize}} \quad & (1-\alpha)\boldsymbol{w}^T\boldsymbol{x} + \frac{\alpha}{2}\boldsymbol{x}^T\mathbf{S}\boldsymbol{x} \\
\text{subject to} \quad & \mathbf{C}\boldsymbol{x} \le \mathbf{1}_{|V_G|+|V_H|} \\
& \boldsymbol{x}(ii') \in \{0,1\}.
\end{aligned} \tag{12}$$

This problem is also equivalent to a binary linear program through a standard linearizing transformation [29], [31]. Different global alignment methods can be viewed as algorithms that either implicitly or explicitly optimize this BQP formulation.

### 3.2 The higher-order generalization

We generalize the node alignment solutions proposed earlier to the problem of aligning higher-order substructures in graphs. As previously mentioned, this is motivated by the existence of motifs in biological networks. In what follows, we assume that the labeling $\ell$ results from a columnwise vectorization of the matrix $\mathbf{X}$ so we can exploit relationships with the Kronecker product. Let $\mathcal{T}_G$ and $\mathcal{T}_H$ be the motif-tensors associated with a motif $\boldsymbol{M}$ in both graphs $G$ and $H$, where this motif has $m$-nodes. Then the higher-order network alignment problem is the binary polynomial problem:

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{maximize}} \quad & (1-\beta)\boldsymbol{w}^T\boldsymbol{x} + \frac{\beta}{m!}(\mathcal{T}_H \otimes \mathcal{T}_G)\boldsymbol{x}^m \\
\text{subject to} \quad & \mathbf{C}\boldsymbol{x} \le \mathbf{1}_{|V_G|+|V_H|} \\
& \boldsymbol{x}(ii') \in \{0,1\}.
\end{aligned} \tag{13}$$

This problem can again be converted into a binary linear program through a standard linearization procedure on the higher-order polynomials, but that is tangential to our discussion here. As a generalization of the network alignment problem, this problem is NP-hard as well.

In this paper, we focus on *triangle* motifs, which are special cases of feed-forward/backward motifs. Please note that our method is general, and can be applied to arbitrary motifs. Denote the triangle tensor of graph $\boldsymbol{G}$ and $\boldsymbol{H}$ using $\triangle_G$ and $\triangle_H$, respectively. Also, denote the triangle tensor for the product graph by $\triangle_{H\times G} = \triangle_H \otimes \triangle_G$. Using this notation, we can write the higher-order network alignment problem as a binary cubic program:

$$\begin{aligned}
\underset{\boldsymbol{x}}{\text{maximize}} \quad & (1-\beta)\boldsymbol{w}^T\boldsymbol{x} + \frac{\beta}{6}(\triangle_{H\times G})\boldsymbol{x}^3 \\
\text{subject to} \quad & \mathbf{C}\boldsymbol{x} \le \mathbf{1}_{|V_G|+|V_H|} \\
& \boldsymbol{x}(ii') \in \{0,1\}.
\end{aligned} \tag{14}$$

In this formulation, $\triangle_{H\times G}\boldsymbol{x}^3$ counts the number of triangles that are conserved under the alignment represented by vector $\boldsymbol{x}$, and $\boldsymbol{w}^T\boldsymbol{x}$ plays a similar role as in BQP formulation of global alignment.

We propose a few heuristic procedures to optimize this objective. First, we remove the one-to-one constraint on $\boldsymbol{x}$ from the optimization problem. Second, we note that the relaxation of this problem over the set of real values coincides with the eigenvectors of tensor $\triangle_{H\times G}$, as presented in Equation 5. In this case, there is a known algorithm, called *shifted symmetric higher-order power method (SS-HOPM)* [13], which can be used to identify eigenpairs of $\triangle_{H\times G}$ with large eigenvalues. Third, we incorporate sequence similarities encoded by $\boldsymbol{w}$ by starting iterations from $\boldsymbol{x}_0 = \boldsymbol{w}$. Finally, we take the real-valued solution from SS-HOPM and form a matrix $\mathbf{X}$ that we use an input to a maximum-weight bipartite matching procedure to generate a one-to-one matching.

We now describe a number of ways to exploit the structure of the problem in this setup for a more efficient computer implementation. The fundamental computational difficulty is manipulating the tensor $\triangle_{H\times G}$. A straightforward application of SS-HOPM on this tensor would utilize a data structure that consumes 55 TB of memory (this exploits sparsity alone, like traditional graph algorithms). To see how this structure balloons in size, consider the case of aligning the yeast and human interactomes. These networks have $347,079$ and $407,650$ triangles, respectively. If we store the non-zeros in a sparse tensor representation of $\triangle_{H\times G}$ using three 32-bit indices per non-zero, it requires (347,079 $\times$ 407,650 motifs) $\times$ (36 symmetry non-zeros per motif) $\times$ (12 bytes per non-zero) $\approx$ 55.5 terabytes of memory to store the product tensor ( or 1.5 terabytes, if we exploit the symmetry). Forming the full non-zero structure, however, is unnecessary as we only need to use this tensor to compute the tensor-times-vector operation, and this can be done implicitly without forming the complete stucture. Before we explain this optimization, we first discuss some of the symmetries in the tensor we can use to reduce storage.

### 3.3 Using symmetry in triangle tensor

We note that there is a high degree of redundancy in a symmetric tensor $\mathcal{T}$. In general, the maximum number of unique elements in a $m^{th}$-order $n$-dimensional tensor, $\mathcal{T} \in \mathbb{R}^{[m,n]}$ is given by [37]:

$$\binom{m+n-1}{m} \quad = \quad \frac{n^m}{m!} + O(n^{m-1}). \tag{15}$$

For the special case of motif-tensors for a graph $\boldsymbol{G}$ with $n$ vertices, we note that all entries for which at least two of the indices are the same would be zero. Each nonzero element in $\mathcal{T}_G$, for which none of $(i_1, \ldots, i_m)$ indices are equal, is repeated exactly $m!$ times. We can use this property to enhance the computation time of $\mathcal{T}_G\boldsymbol{x}^k$ and $\mathcal{T}_G\boldsymbol{x}^{m-1}$, which are basic kernels in SS-HOPM, by computing each redundant element only once and adjusting the sum by the number of repetition. Specifically, when computing $\mathcal{T}\boldsymbol{x}^{m-1}$, we have:

$$(\mathcal{T}\boldsymbol{x}^{m-1})_{i_1} = \sum_{i_2,\ldots,i_m} \mathcal{T}_{i_1,\ldots,i_m}\boldsymbol{x}_{i_2}\cdot\boldsymbol{x}_{i_3}\cdot\ldots\cdot\boldsymbol{x}_{i_m}. \tag{16}$$

However, any permutation of indices $i_2, \ldots, i_m \in \Pi_{m-1}$ yields an identical term in this summation. To use this property, we can define a canonical labeling for $i_2, \ldots, i_m$ and compute it only once, and then multiply the overall sum by $(m-1)!$. We will use this property to further reduce the computation time of our algorithm for the special case of triangle tensors.

### 3.4 An implicit kernel for computing tensor-vector products

Before we present the full TAME algorithm, we need one additional building block for computing the tensor-vector product. The key challenge is that the number of elements in the triangle tensor of product graph, $\triangle_{H\times G}$, is too large to fit in the memory of most modern computers, even for relatively small graphs. To remedy this problem, we note that there is no need to explicitly construct $\triangle_{H\times G}$. All we need to run SS-HOPM is to compute $\triangle_{H\times G}\boldsymbol{x}^3$ and $\triangle_{H\times G}\boldsymbol{x}^2$. Rewriting the tensor-vector product formulation in Equation 2, we find the following vertex-centered, implicit kernel as follows:

$$\begin{aligned}
&(\triangle_{H\times G}\boldsymbol{x}^2)_{ii'} \\
&\quad= \sum_{jj',kk'} \triangle_{H\times G}(ii',jj',kk')\boldsymbol{x}(jj')\boldsymbol{x}(kk') \\
&\quad= \sum_{j,j',k,k'} \triangle_G(i,j,k)\triangle_H(i',j',k')\mathbf{X}(j,j')\mathbf{X}(k,k') \\
&\quad= \sum_{j,k} \triangle_G(i,j,k)\sum_{j'}\mathbf{X}(j,j')\sum_{k'}\triangle_H(i',j',k')\mathbf{X}(k,k')
\end{aligned} \tag{17}$$

where $\mathbf{X} = \mathbf{unvec}(\boldsymbol{x})$. Additionally, we can simplify this vertex-centered formulation to derive a more efficient motif-centered kernel. To this end, we note that triangle tensors of graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ represent a 3-uniform hypergraph over the set of vertices $V_G$ and $V_H$, respectively. Denote the hypergraph incidences of these hypergraphs by $\mathcal{N}_{\triangle_G}$ and $\mathcal{N}_{\triangle_H}$, where $\mathcal{N}_{\triangle_G}(i) = \{(j,k) \mid (v_i,v_j),(v_j,v_k),(v_k,v_i) \in E_G\}$, and $\mathcal{N}_{\triangle_H}(i') = \{(j',k') \mid (v_{i'},v_{j'}),(v_{j'},v_{k'}),(v_{k'},v_{i'}) \in E_H\}$. Then

$$\begin{aligned}
&\triangle_{H\times G}\boldsymbol{x}^2(ii') = \\
&\quad 2\sum_{(j,k)\in\mathcal{N}_{\triangle_G}(i)} \sum_{(j',k')\in\mathcal{N}_{\triangle_H}(i')} \mathbf{X}(j,j')\mathbf{X}(k,k')+\mathbf{X}(j,k')\mathbf{X}(k,j').
\end{aligned} \tag{18}$$

In this formulation, we also make use of the symmetric property of triangle tensors. The outer factor of 2 corresponds to the $(m-1)$ degree of symmetries, introduced in Section 3.3. The inner summation accounts for the fact that nodes $v_j$ and $v_k$ from $\boldsymbol{G}$, or vertices $\{v_{i_2}, \ldots, v_{i_k}\}$ when dealing with motif-tensors of size $k$, can be mapped to their counterpart vertices $v_{j'}$ and $v_{k'}$ in $\boldsymbol{H}$ in $(m-1)!$ different ways. Each of these mappings contribute a factor of one in $\triangle_{H\times G}$. However, their corresponding $\boldsymbol{x}$ values are different and we need to separately compute their product. We use this motif-centered formulation in our final algorithm to compute the implicit tensor-kernel product, $\tilde{\boldsymbol{x}} = \triangle_{H\times G}\boldsymbol{x}^2$. Having $\tilde{\boldsymbol{x}}$, one can easily compute $\triangle_{H\times G}\boldsymbol{x}^3 = \boldsymbol{x}^T\tilde{\boldsymbol{x}}$. The simplified pseudo-code of the implicit kernel for computing $\triangle_{G\times H}\boldsymbol{x}^2$ is provided in Algorithm 1. The computation time of this algorithm is of $O(|\triangle_G| \times |\triangle_H|)$.

### 3.5 Triangular AlignMEnt (TAME) algorithm

We now integrate different building blocks introduced earlier to present a higher-order alignment method for trian-

**Algorithm 1** Implicit tensor-times-vector product (impTTV)

**Input:** Triangle-tensors $\triangle_G, \triangle_H$, for $\boldsymbol{G}$ and $\boldsymbol{H}$; a vector $\boldsymbol{x}$
**Output:** $\boldsymbol{y} = \triangle_{H \times G} \boldsymbol{x}^2$

$\mathbf{X} = \mathbf{unvec}(\boldsymbol{x})$
$\mathbf{Y} = \mathbf{0}$
**for** $v_i \in V_G$ **do**
  **for** $v_{i'} \in V_H$ **do**
    **for** $\{(j,k) \in N_{\triangle_G}(i)\}$ **do**
      **for** $\{(j',k') \in N_{\triangle_H}(i')\}$ **do**
        $\mathbf{Y}(i,i')$+=$\mathbf{X}(j,j')\mathbf{X}(k,k') + \mathbf{X}(j,k')\mathbf{X}(k,j')$
      **end for**
    **end for**
    $\mathbf{Y}(i,i') = \mathbf{Y}(i,i') * 2$
  **end for**
**end for**
$\boldsymbol{y} = \mathbf{vec}(\mathbf{Y})$

gle motifs. This code uses one additional primitive. The function **score** solves a bipartite maximum-weight matching problem (using the Hungarian method) and returns the total number of triangles $t$ aligned by the matching. The pseudocode for TAME algorithm is presented in Algorithm 2. The final alignment results from running one additional maximum-weight bipartite matching on the returned topological similarity scores $\mathbf{X}$.

**Algorithm 2** The Triangular AlignMEnt (TAME) algorithm

**Input:** Triangle tensors $\triangle_G, \triangle_H$; Sequence similarities $\boldsymbol{w}$; Shift parameter $\alpha$
**Output:** The best topological scores $\mathbf{X}$ from any iteration

1: $k = 0$ {Iteration number}
2: $\boldsymbol{w} \leftarrow \boldsymbol{w}/\|\boldsymbol{w}\|$
3: $\boldsymbol{x_0} = \boldsymbol{w}$
4: $t_0 = 0$
5: **repeat**
6: $\tilde{\boldsymbol{x}}_{k+1} = \mathbf{impTTV}(\triangle_G, \triangle_H, \boldsymbol{x}_k)$
7: $\lambda_{k+1} = \boldsymbol{x}_k^T \tilde{\boldsymbol{x}}_{k+1}$
8: $\hat{\boldsymbol{x}}_{k+1} = \tilde{\boldsymbol{x}}_{k+1} + \alpha \boldsymbol{x}_k$
9: $\boldsymbol{x}_{k+1} = \frac{\hat{\boldsymbol{x}}_{k+1}}{\|\hat{\boldsymbol{x}}_{k+1}\|}$
10: $\mathbf{X}_{k+1} = \mathbf{unvec}(\boldsymbol{x_{k+1}})$
11: $t_{k+1} = \mathbf{score}(\mathbf{X_{k+1}})$
12: Update $(\mathbf{X}, t)_{\text{best}}$ to $(\mathbf{X}, t)_{k+1}$ if $t_{k+1} > t_{\text{best}}$
13: $k = k + 1$
14: **until** $\lambda_k - \lambda_{k-1}$ is small or the max iteration is hit
15: **return** $\mathbf{X}_{\text{best}}$

The overall algorithm takes in the prior similarity and uses that as the starting interation of the SS-HOPM process (lines 5-14). In that process, **impTTV** uses the motif-centered, implicit tensor-times-vector kernel proposed in Section 3.4. The SS-HOPM loop generates a sequence of topological similarity matrices. However, the SS-HOPM process generates a sequence to optimize the problem after removing the two constraints in Equation 14, namely the integer constraint on $\boldsymbol{x}$ and the one-to-one matching constraint over $\mathbf{X}$. To enforce these constraints, we perform a matching in each iteration of the algorithm and generate a score on the iterate that returns the aligned triangle count. We keep the highest scoring topological matrix as $\mathbf{X}_{\text{best}}$.

In addition to Algorithm 2, which we refer to as **full TAME**, we present a variant of this algorithm, called **contrained TAME**, which only matches nodes that have *at least one match* suggested by the prior alignment. In this formulation, we must update $\triangle_G$ and $\triangle_H$ prior to running the full TAME algorithm. The key idea is to remove triangles for which at least one of the end-points has no homology suggested by the sequence similarity. This allows us to focus on the most promising regions of the graph. The contrained TAME method is presented in Algorithm 3. In this algorithm, we first compute a pair of indicator vectors, $\boldsymbol{w}_r$ and $\boldsymbol{w}_c$ with size $|V_G|$ and $|V_H|$, respectively. Each element $i$ in $\boldsymbol{w}_r$ indicates if vertex $v_i \in V_G$ has at least one homolog among vertices of $\boldsymbol{H}$ and, similarly, each element $i'$ in $\boldsymbol{w}_c$ indicates if vertex $v_{i'} \in V_H$ has at least one homolog among vertices of $\boldsymbol{G}$ (determined by the prior similarity). The "**.***" operator is the element-wise product of two tensors. Finally, we prune the triangle tensors by enforcing that all end-points of every triangle motif should have at least one homolog in the other graph. An equivalent way of understanding this algorithm is that we first remove vertices from $\boldsymbol{G}$ and $\boldsymbol{H}$ that have no prior information indicating there is a match in the other graph.

**Algorithm 3** The contrained Triangular AlignMEnt (cTAME) algorithm

**Input:** Triangle tensors $\triangle_G, \triangle_H$; Sequence similarities $\boldsymbol{w}$; Shift parameter $\alpha$
**Output:** The final set of aligned node pairs $\langle \boldsymbol{m_i}, \boldsymbol{m'_i} \rangle$

$\mathbf{W} = \mathbf{unvec}(\boldsymbol{w})$
$\boldsymbol{w}_G =$ indicator vector for rows of $\mathbf{W}$ with non-zeros
$\boldsymbol{w}_H =$ indicator vector for cols of $\mathbf{W}$ with non-zeros
$\mathcal{W}_G = \boldsymbol{w}_G \otimes \boldsymbol{w}_G \otimes \boldsymbol{w}_G$
$\mathcal{W}_H = \boldsymbol{w}_H \otimes \boldsymbol{w}_H \otimes \boldsymbol{w}_H$
$\triangle_G^{(\text{contrained})} = \triangle_G . * \mathcal{W}_G$
$\triangle_H^{(\text{contrained})} = \triangle_H . * \mathcal{W}_H$
$\mathbf{X} = \mathbf{TAME}(\triangle_G^{(\text{contrained})}, \triangle_H^{(\text{contrained})}, \boldsymbol{w}, \alpha)$

This algorithm has the side-effect of reducing the total number of triangles (see Table 2), resulting in a faster execution time. In many cases, it also outperforms the full version of TAME in terms of alignment quality by focusing the search in more promising regions. We discuss the pros and cons of each of these methods in Section 4

## 4 Results and Discussion

### 4.1 Datasets

#### 4.1.1 Synthetic Datasets and Random Networks

NAPAbench [16], is a family of random graphs that has been proposed for evaluating network alignment methods on synthetic datasets. This dataset contains both pairs of networks for evaluating pairwise alignment methods, as well as group of networks for testing multiple alignment algorithms. There are three random graph generation models employed by NAPAbench: (i) duplication-mutation-complementation (DMC), (ii) duplication with random mutation (DMR), and (iii) crystal growth (CG). These random networks mimic key properties of biological graphs, including their network topology and modular structure.

TABLE 1
Summary statistics for the NAPAbench dataset

| | # nodes | Mean # edges | Mean # triangles |
|---|---|---|---|
| Graph A | 3,000 | 11,985 | 11,362 |
| Graph B | 4,000 | 15,985 | 15,880 |

TABLE 2
Summary statistics for yeast and human interactomes.

| | # nodes | # edges | # triangles |
|---|---|---|---|
| Human | 14,867 | 126,593 | 407,650 |
| Yeast | 5,850 | 79,458 | 347,079 |
| constrained Human | 10,624 | 88,276 | 251,555 |
| constrained Yeast | 5,482 | 73,739 | 289,893 |

We focus on the crystal growth (CG) dataset, which is based on a newer model that better fits features of real PPI networks, including their characteristic age distribution [38]. This dataset contains 10 pairs of graphs, for which the known orthology and simulated sequence similarities between pairs of nodes are available. Each pair consists of a first graph $A$ with $3,000$ nodes and a second graph $B$ with $4,000$ nodes. These two networks share a common ancestor of size $2,000$ and have been evolved independently after that. Edge and triangle statistics for these graphs are summarized in Table 1

#### 4.1.2 Yeast Versus Human Interactome Dataset

Both yeast and human protein-protein interaction (PPI) networks were constructed from BioGRID database, version 3.2.103. All physical interactions, excluding self-loops and interspecies interactions, have been filtered and mapped to Entrez gene IDs. We used these interaction evidences to construct the adjacency matrix for both graphs. Edge and triangle statistics for each network are presented in Table 2.

We downloaded the protein sequences for the yeast and humans genes in FASTA format from Ensembl database, release 69. These datasets are based on the GRCh37 and EF4 reference genomes, each of which contain 101,075 and 6,692 protein sequences for *H. Sapiens* and *S. Cerevisiae*, respectively. Each human gene in this dataset has, on average, around 4 protein isoforms. We identified and masked low-complexity regions in protein sequences using *pseg* program [39]. The *ssearch36* tool, from *FASTA* [40] version 36, was then used to compute the local sequence alignment of the protein pairs using the Smith-Waterman algorithm [41]. We used this tool with the BLOSUM50 scoring matrix to compute sequence similarity of protein pairs in humans and yeast. All sequences with E-values less than or equal to 10 are recorded as possible matches, which results in a total of 664,769 hits between yeast and human proteins. For genes with multiple protein isoforms, coming from alternatively spliced variants of the same gene, we only record the most significant hit. The final dataset contains 162,981 pairs of similar protein-coding genes. After mapping these pairs to the human and yeast interactomes, we were able to find matches for 127,505 node pairs in these networks.

### 4.2 Benchmark Methods

To evaluate TAME, we compare it against three different methods: (i) IsoRank [28], (ii) Belief Propagation (BP) [42], and (iii) GHOST [36]. We use IsoRank because it is one of the most widely used global network aligners in the literature. We use GHOST due to its similarity with full version of TAME in computing topological similarities independently from sequence similarities. We include BP in our comparison because it optimizes the alignment procedure by truncating the search space to only subsets of possible homologs. Additionally, we tried H-GRAAL [33] as it provides topological similarities similar in nature to TAME and GHOST; however, the graphlet signature computation phase had an estimated time of more than a year. Thus, we had to exclude this algorithm from our experiments.

For a fair comparison of different alignment methods, we treat each method as producing a real-valued similarity matrix $\mathbf{S}(i, i')$ between the nodes of $\boldsymbol{G}$ and $\boldsymbol{H}$. This separates the computation phase from the matching phase, as also proposed by Kollias *et al.* [43]. We use the Hungarian maximum weight bipartite matching algorithm as the final matching block for all methods. The details of how this works is explained below for each of the methods.

We use the versions of IsoRank and Belief Propagation as described and implemented in Bayati et al. [31]. IsoRank runs iterations of the PageRank method on the Kronecker product graph $A_{\boldsymbol{H}} \otimes A_{\boldsymbol{G}}$ using a seed or localization vector derived by normalizing the sequence similarities to a probability distribution. At each step, the method solves a maximum weight bipartite matching problem to score the current heuristic alignment and picks the best score from any iteration, just like TAME. The BP method runs an iterative message passing algorithm on a graphical model that represents the network alignment problem restricted to the set of potential matches induced by the sequence similarity matrix $\mathbf{W}$. Again, the method solves a maximum weight bipartite matching problem at each step to score the current iterate and picks the best.

For GHOST, we provide some details on the method in the appendix because our implementation differs from that given by the authors' software. We modified the method to establish a fair comparison between the amount of topological methods the various methods extract. The authors of GHOST spent considerable effort designing a custom procedure tuned to the nature of their topological information to identify an accurate one-to-one matching. The point of our experiments is that TAME provides rich topological information with a simple matching strategy. Consequently, we use the GHOST procedure to estimate a topological similarity matrix $\mathbf{S}_{\text{topo}}(i, i')$ with large entries if there is a high topological similarity between vertex $v_i \in V_G$ and $v_{i'} \in V_H$. We then use a convex combination of topological and sequence similarity to procedure a network alignment heuristic:

$$\mathbf{X}_{\text{GHOST-Topo}}(i, i') = \alpha \mathbf{S}_{\text{topo}}(i, i') + (1 - \alpha)\mathbf{W}(i, i'). \quad (19)$$

As with the other methods, we solve a maximum weight bipartite matching problem on $\mathbf{X}_{\text{GHOST-Topo}}(i, i')$ to produce the alignment.

### 4.3 Evaluation Criteria

For each resulting alignment, we separately assess the topological quality of the alignment graph, as well as the biological relevance of aligned nodes in the input graphs. Additionally, for the NAPAbench synthetic dataset, we directly compute the percent of correctly aligned node pairs. Let $m(ii')$ be an indicator to denote that vertex $v_i \in V_G$ is matched to $v_{i'}$ in $V_H$. The description of different performance measures are as follows:

#### 4.3.1 Node Correctness (NC)

This measure is only defined for synthetic cases for which the true-alignment is known a priori. For NAPAbench, there is a shared ancestor core of $2,000$ nodes between paired set of networks. In this case, the node correctness is defined as the percent of these $2,000$ node pairs that are correctly aligned by each alignment method.

#### 4.3.2 Edge Correctness (EC)

After constructing the alignment graph, we can compute the total number of conserved edges from the edge-set of the alignment graph, i.e. $E_A = \{(ii', jj') \mid (i,j) \in E_G, (i',j') \in E_H,$ **and** $m(ii') = m(jj') = 1\}$. After normalizing this count, we can define edge correctness as follows:

$$EC = 100 \cdot \frac{|E_A|}{\mathbf{min}(|E_G|, |E_H|)}. \tag{20}$$

Note that despite the name, edge correctness only measures the fraction of possible edges aligned, not the fraction of accurately aligned edges.

#### 4.3.3 Triangle Correctness (TC)

Similar to edge correctness, triangle correctness is defined on the basis of total number of conserved triangles. It can be represented with respect to the triangle-set of the alignment graph, $T_A = \{(ii', jj', kk') \mid (i,j,k) \in T_G, (i',j',k') \in T_H,$ **and** $m(ii') = m(jj') = m(kk') = 1\}$. After normalizing this count, we can define triangle correctness as follows:

$$TC = 100 \cdot \frac{|T_A|}{\mathbf{min}(|T_G|, |T_H|)}. \tag{21}$$

#### 4.3.4 Ortholog Correctness (OC)

When the true alignment is not known, we cannot use node correctness to directly assess the quality of aligned pairs. Instead, we need to use other measures as a proxy for potential ortholog pairs. Here, we use Gene Ontology (GO) [44] to evaluate matches. We separately evaluate based on three independent branches of GO, namely Biological Process (BP), Molecular Function (MF), and Cellular Component (CC). To avoid over-counting terms, we follow suggestions in the literature [45] and focus on the annotations that are at the shortest path distance of 5 from the root of GO hierarchy. Moreover, to avoid terms that are predicted based on the sequence similarity, we only include the set of experimental annotations, namely terms with evidence codes `EXP`, `IDA`, `IPI`, `IMP`, `IGI`, and `IEP`. The final dataset includes $15,595$ BP terms, $4,221$ MF terms, and $2,761$ CC terms.

To assess a pair of aligned nodes $m(ii') = 1$, we first identify the annotation sets: $F_{\text{BP}}(v_i), F_{\text{MF}}(v_i), F_{\text{CC}}(v_i)$ for $v_i$ and $F_{\text{BP}}(v_{i'}), F_{\text{MF}}(v_{i'}), F_{\text{CC}}(v_{i'})$ for $v_{i'}$. Then, we score the alignment pairs for a fixed branch of GO by evaluating the significance of the observed overlap among the annotations of $v_i$ and $v_{i'}$, quantified as $O = |F(v_i) \cap F(v_{i'})|$. Instead of using the Jaccard index, which only considers the size of union and intersection sets, we propose a statistical test based on the hypergeometric test.

In this formulation, the null model is that annotations are drawn at random from the universe of terms. Let $U$ be the number of terms for the current comparision (for instance, $U = 2,761$ if we are scoring based on the CC terms). Let $P = |F(v_i)|$ and $Q = |F(v_{i'})|$. We need to assess the probability that a random set of $P$ terms drawn without replacement from the universe set $U$ has $O$ or more terms in common with the set $F(v_{i'})$. The hypergeometric distribution provides a means to compute these probabilities. Let $HG_{\text{pdf}}(x|U,P,Q)$, for $O \le x \le Q$, be the probability of seeing an intersection of size $x$. Using this notation, the overall $p$-value can be computed as:

$$p\text{-value}(O) = \sum_{x=O}^{\min(P,Q)} HG_{\text{pdf}}(x|U,P,Q). \tag{22}$$

Note that this $p$-value is specific to a single pair $(i,i')$, and thus, $O, P,$ and $Q$ need to be determined by that context. Using the Bonferroni correction method to correct for multiple hypothesis testing, we can identify the final set of ortholog pairs as $O_A = \{ii' \mid p\text{-value}(O_{ii'}) \le \tau/|E_A|$ **and** $m(ii') = 1\}$. The ortholog correctness score then reports the fraction of matches with biological significance given the GO annotations:

$$OC = 100 \cdot \frac{|O_A|}{|E_A|}. \tag{23}$$

In all our experiments, we used a threshold of $\tau = 0.05$ to declare a $p$-value as significant.

### 4.4 Experimental settings

#### 4.4.1 Parameter tunning

Both the IsoRank and the BP alignment methods depend on a parameter to control the weight of sequence similarity and topological similarity in their iterative algorithms. Also, for GHOST, we can tune these weights based on the final combination. We evaluate these methods with:

| | |
|---|---|
| IsoRank | $1-\alpha$ is the sequence weight<br>$\alpha$ is the topological weight<br>$\alpha = 0.15, 0.5, 0.85$ |
| BP | $\alpha$ is the sequence weight<br>$\beta$ is topological weight<br>$(\alpha,\beta) = (0.15, 0.85), (0.5, 0.5), (0.85, 0.15)$ |
| GHOST | $1-\alpha$ is the sequence weight<br>$\alpha$ is the topological weight<br>$\alpha = 0.85, 0.5, 0.85, 1$ |

(Note that $\alpha$ means something slightly different for each algorithm.)

These points span a range between low and high topological influence (and high to low sequence influence). For TAME, we only have the $\alpha$ parameter that corresponds to the extent of included shift. To tune this parameter, we run the algorithm using values of the shift parameter over a log-linear search space ($\alpha \in$

$\{0, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}\}$) and choose the maximum based on the number of aligned triangles. We only report the value of the shift parameter with the best performance. The contrained and full formulations of TAME have greatly different nonzeros in the product tensors. To this end, we run the parameter tuning phase for each of them independently. In case of a random graph ensemble in NAPAbench, we compute the optimal shift values for each pair of networks, independently, and then use a majority voting technique to identify the value that performs the best in majority of alignments. More recent methods such as the generalized eigenproblem adaptive power (GEAP) [46] have been proposed as an extension of *SS-HOPM* that can identify an adaptive shift in each iteration using the Hessian matrix of tensor-vector product.

#### 4.4.2 Implementation

We have implemented both sparse and full version of the TAME algorithm in C/C++, which is accessible from https://github.com/shmohammadi86/TAME. Additionally, we implemented GHOST topological similarity computation in Matlab. For Belief Propagation and IsoRank algorithms, we used the NetAlign Matlab package which is implemented by one of the authors [31]. Finally, we packaged all Matlab codes, together with the evaluation codes/datasets, which is available for download from https://github.com/shmohammadi86/HigherOrder-NetAlign.

### 4.5 NAPABench evaluation

We aligned each pair of networks (a total of ten) separately using the different alignment methods. The tuned value of $\alpha$ for cTAME was $0$ and for TAME it was $10^{-1}$, based on some simple tie-breaking rules. Table 3 summarizes each of the evaluation measures, and provides their mean and standard deviation across all aligned networks. Methods in this table are sorted alphabetically.

In Figure 1(a), we display a list of methods sorted by their node correctness score to illustrate how the various methods compare in terms of number of correctly aligned nodes. In Figure 1(b), we order the methods based on their aggregate edge and triangle correctness scores. This illustrates how the methods compare in terms of the aligned topology of the graphs. The TAME algorithm performed the best across all three measures. The second-best method is the BP algorithm. This method enjoys a substantial advantage in that it cannot generate any new homolog information beyond what is given by sequence similarities. For the NAPAbench experiments, this is guaranteed to contain the correct alignment. The poor performance of GHOST is understandable in this setting because it uses eigenvalue distributions of local subgraphs. Eigenvalue distributions of *random graphs* typically follow various well-known distributions. For instance, the eigenvalues of an Erdős-Rényi graph obey the Wigner semi-circle law (with one outlier). It is likely that because of the random graphs generated in NAPAbench, the local subgraph neighborhood densities follow similar patterns, thus making it almost impossible for GHOST to exploit the spectral signatures. We believe it is likely the optimized matching routine in the authors' implementation of the GHOST method is able to work around these difficulties.

### 4.6 Alignment of human versus yeast interactomes

To assess the performance of different alignment methods when applied to real networks, we ran experiments on the yeast and human interactomes. The optimal value of $\alpha$ for the shift parameter in TAME and cTAME are $10$ and $0.1$, respectively. Recall that we cannot compute node correctness in this case as the true orthologs are unknown. Thus, in this case, we treat the ortholog correctness scores as a proxy.

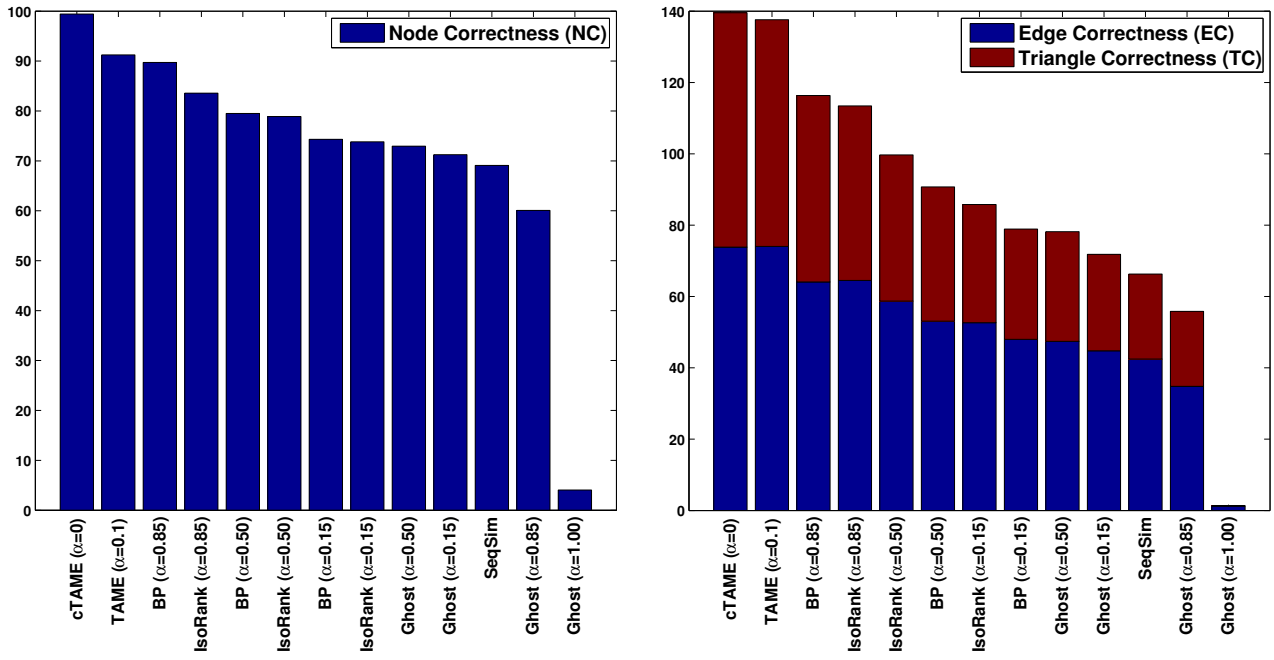
Table 4 summarizes the various measures computed for each alignment method. These methods are sorted alphabetically, and corresponding columns are related to topological (edge and triangle correctness) and biological measures (ortholog correctness) of the alignment. Figures 2(a) and 2(b) illustrates the aggregate biological and topological measures sorted individually based on their performance. Unlike NAPAbench we observe a considerable difference among the methods on these two evaluation components.

In terms of topological correctness, the cTAME method is among the worst and the TAME method is the best by a considerable margin. In contrast, for the ortholog correctness measures, the cTAME method is slightly better than the various results produced by BP, which is turn, is only slightly better than sequence similarity alone (recall that BP benefits from a substantial advantage in the reduced search space). The cTAME method outperformed all of the other algorithmic methods on each of the ortholog correctness measures, which suggests that triangles in general are more informative than edges when aligning networks. Additionally, both BP and cTAME outperform pure sequence similarities, which means that using topology can effectively differentiate between potential orthologs.

We have a number of potential explanations for the observed disparity between the ortholog and the topological evaluations. Our results indicate that there is a large region of the PPIs that is topologically similar between the human and yeast (and found by TAME), which is not present among the homologs (and hence, missed by cTAME). The existence of this large, triangle-enriched core can be an artifact of including many false positive edges in the graphs $\boldsymbol{G}$ and $\boldsymbol{H}$. These results also indicate that TAME is particularly sensitive to *correlated* errors in the network. The issue with correlated errors is that the number of possible triangles induced by false edges grows (at worst) cubically with the size of a mismatched region of the graph, whereas the number of edges grows (at worst) quadratically. Hence, it is possible for a small set of correlated mistakes to greatly impact the solutions of TAME. Pruning edges that are incident to nodes with no known homology is a stringent measure that reduces this possibility. Another possible cause can be attributed to the large evolutionary distance between human and yeast, resulting in divergence of a large number of homologous genes, for which topological clues may be still conserved. Finally, we note that there is a significant bias in the GO annotations towards homolog pairs and these pairs are more extensively studied in the literature due to their importance. To this end, cTAME does not identify *de novo* orthologs without sequence similarity evidence, whereas TAME is capable of reporting missing ortholog pairs.

TABLE 3
Summary statistics for different alignment methods applied to the NAPAbench synthetic random networks (sorted alphabetically)

| Method | Node Correctness (NC) | Edge Correctness (EC) | Triangle Correctness (TC) |
|---|---|---|---|
| BP ($\alpha$=0.15) | 74.32 $\pm$ 8.76 | 47.99 $\pm$ 9.42 | 30.93 $\pm$ 12.19 |
| BP ($\alpha$=0.50) | 79.49 $\pm$ 6.93 | 53.07 $\pm$ 7.67 | 37.65 $\pm$ 9.90 |
| BP ($\alpha$=0.85) | 89.72 $\pm$ 3.35 | 64.08 $\pm$ 3.77 | 52.27 $\pm$ 4.80 |
| Ghost ($\alpha$=0.15) | 71.25 $\pm$ 1.69 | 44.79 $\pm$ 1.88 | 27.02 $\pm$ 2.34 |
| Ghost ($\alpha$=0.50) | 72.94 $\pm$ 1.55 | 47.42 $\pm$ 1.70 | 30.75 $\pm$ 2.05 |
| Ghost ($\alpha$=0.85) | 60.08 $\pm$ 1.38 | 34.81 $\pm$ 1.49 | 21.03 $\pm$ 1.21 |
| Ghost ($\alpha$=1.00) | 4.05 $\pm$ 0.57 | 1.27 $\pm$ 0.32 | 0.04 $\pm$ 0.03 |
| IsoRank ($\alpha$=0.15) | 73.83 $\pm$ 1.49 | 52.66 $\pm$ 1.98 | 33.13 $\pm$ 2.11 |
| IsoRank ($\alpha$=0.50) | 78.88 $\pm$ 1.28 | 58.70 $\pm$ 1.61 | 40.98 $\pm$ 1.99 |
| IsoRank ($\alpha$=0.85) | 83.58 $\pm$ 0.98 | 64.52 $\pm$ 1.23 | 48.91 $\pm$ 1.75 |
| SeqSim | 69.09 $\pm$ 1.81 | 42.47 $\pm$ 1.97 | 23.83 $\pm$ 2.26 |
| cTAME ($\alpha$=0) | **99.42 $\pm$ 0.13** | 73.85 $\pm$ 0.27 | **65.69 $\pm$ 0.31** |
| TAME ($\alpha$=1) | 91.23 $\pm$ 0.99 | **74.03 $\pm$ 0.99** | 63.56 $\pm$ 1.26 |




(a) Node correctness (b) Topological correctness

Fig. 1. Comparison of alignment quality on NAPAbench synthetic dataset based on the mean quality from 10 networks.

### 4.7 TAME's iteration behavior

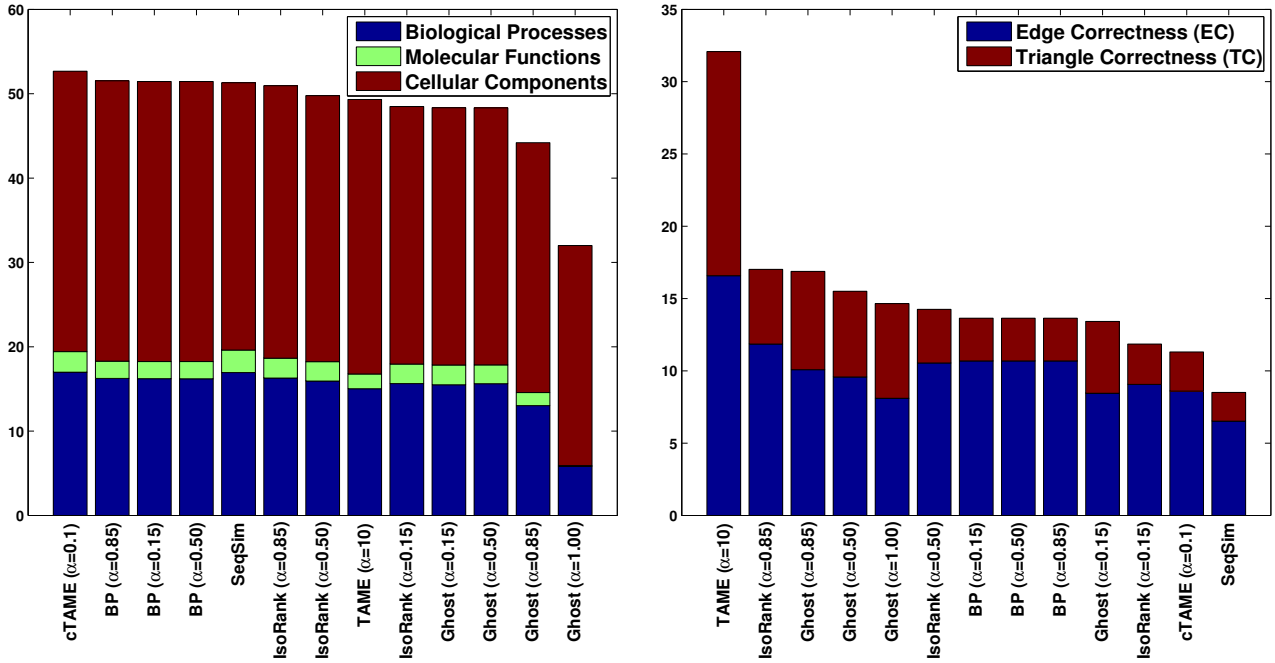
We now make a few comments on the various iterations produced by the TAME method. A curious behavior of TAME is that in almost all cases, the best solution occured within the first few iterations. The same characteristic is observed both for aligning the NAPAbench and real PPI networks. We note that since SS-HOPM does not have any means to internally avoid many-to-many mappings, the dominant eigenvector of $\mathcal{T}$ has a unique structure in which every node in one graph points to the most promising nodes in the other graph. In order to visualize this characteristic, we ran TAME over the Family 1 dataset in NAPAbench and visualized the structure of similarity matrix in each iteration. Figure 3 illustrates the first 15 iterations of the algorithm. We permuted rows and columns to highlight the orthologies as the diagonal of the matrix. As such, the iterations start with all sequence similarities scattered around diagonal elements (Iteration 1), and many false positive off-diagonal pairs. As iterations continue, we start by finding a block diagonal structure (Iterations 2-5), representing triangle enriched regions in the networks. As the process further continues, one of the blocks emerges as the stationary point (Iterations 6-11). Subsequent iterations localize around a solution induced by this block (Iterations 12-15). We are currently seeking theoretical characterizations of this behavior that may suggest improved methods. For instance, it would be useful to avoid the transition to only one block that occurs during Iterations 6-11.

## 5 Concluding Remarks and Future Work

In this paper we propose an alternative formulation for the network alignment problem, which allows us to use higher-

TABLE 4
Summary statistics for different alignment methods applied to the yeast and human interactomes (sorted alphabetically)

| Method | Edge Correctness | Triangle Correctness | *Ortholog Correctness* Bio. Processes | *Ortholog Correctness* Mol. Functions | *Ortholog Correctness* Cell. Components |
|---|---|---|---|---|---|
| BP ($\alpha$=0.15) | 10.69 | 2.95 | 16.22 | 2.03 | 33.20 |
| BP ($\alpha$=0.50) | 10.69 | 2.96 | 16.20 | 2.04 | 33.21 |
| BP ($\alpha$=0.85) | 10.68 | 2.95 | 16.24 | 2.05 | **33.24** |
| Ghost ($\alpha$=0.15) | 8.45 | 4.97 | 15.49 | 2.34 | 30.51 |
| Ghost ($\alpha$=0.50) | 9.58 | 5.93 | 15.62 | 2.22 | 30.48 |
| Ghost ($\alpha$=0.85) | 10.08 | 6.80 | 13.03 | 1.54 | 29.62 |
| Ghost ($\alpha$=1.00) | 8.10 | 6.55 | 5.81 | 0.10 | 26.09 |
| IsoRank ($\alpha$=0.15) | 9.06 | 2.80 | 15.64 | 2.31 | 30.53 |
| IsoRank ($\alpha$=0.50) | 10.55 | 3.71 | 15.93 | 2.31 | 31.54 |
| IsoRank ($\alpha$=0.85) | 11.86 | 5.16 | 16.29 | 2.36 | 32.31 |
| SeqSim | 6.52 | 1.99 | 16.94 | **2.66** | 31.70 |
| cTAME ($\alpha$=0.1) | 8.60 | 2.71 | **16.98** | 2.45 | **33.24** |
| TAME ($\alpha$=10) | **16.58** | **15.50** | 15.03 | 1.74 | 32.55 |




(a) Ortholog correctness (b) Topological correctness

Fig. 2. Comparison of alignment quality on NAPAbench synthetic dataset based on the mean quality from 10 networks.

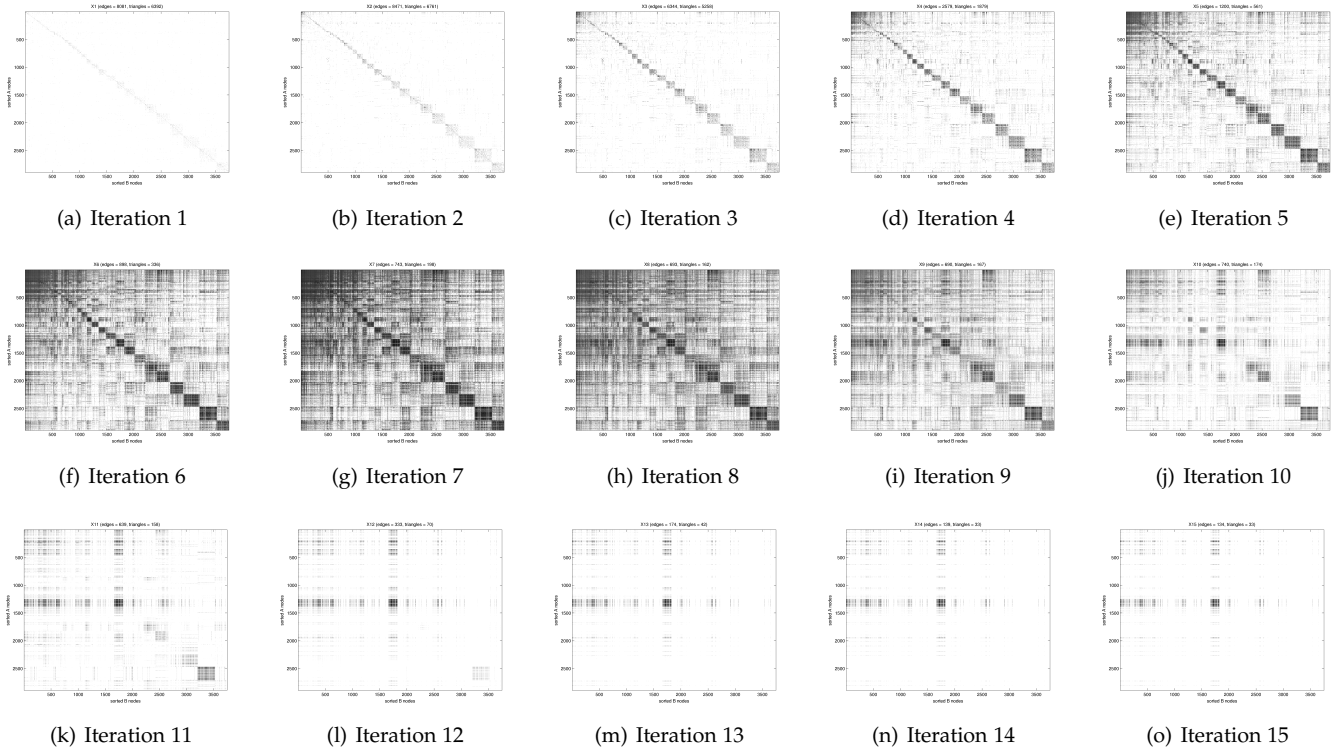
order substructures to drive the alignment process. We provide the necessary machinery to encode different motifs using tensors; and, as a proof of concept, we use triangle motifs to show how the framework can be applied to the network alignment problem. We show that our method outperforms state of the art techniques, both in the context of real and synthetic datasets.

The result of our method is a set of topological scores that can be combined with many of the other ideas in the network alignment literature. For instance, the information contained in the TAME iterates seems to be orthogonal to the information produced by a method such as GHOST. We believe it is likely that these different similarity scores can be integrated—perhaps by using local features of the graph topology to indicate which is more reliable—and the result should be better than either.

Our ongoing work is focused on optimizing the implicit kernel, enhancing mixing properties of sequence and topological similarities, extending the main iteration to simultaneous subspace iteration with nonnegative orthogonalization, combining motifs of different sizes into the optimization problem, and understanding the theoretical basis of the success of the early iterations.

## APPENDIX

Patro *et al.* [36] have proposed an algorithm for global network alignment, called GHOST, that consist of two main components: (i) a novel topological similarity score, (ii) a greedy search algorithm to identify best matches considering sequence and topological similarities. Here, we focus



(a) Iteration 1 (b) Iteration 2 (c) Iteration 3 (d) Iteration 4 (e) Iteration 5

(f) Iteration 6 (g) Iteration 7 (h) Iteration 8 (i) Iteration 9 (j) Iteration 10

(k) Iteration 11 (l) Iteration 12 (m) Iteration 13 (n) Iteration 14 (o) Iteration 15

Fig. 3. The first 15 iterations of TAME applied to Family_1 in the NAPAbench illustrated as a matrix plot of iterates $\boldsymbol{x}$ reshaped to a $\mathbf{X}$ where the true orthologs lie on the diagonal. These illustrate how the best alignments result from the information in the first few (2-5) iterations.

on the former component, as we use maximum bipartite matching for all alignment algorithms to provide a fair comparison. The idea behind the GHOST similarity score is to define a spectral signature for each node, which can be later used to assess their topological distances.

To this end, they first construct induced subgraphs of different radii up to distance $k$ around each node, where $k$ is the calculated based on shortest path distance in the undirected, unweighted graphs of protein-protein interactions. Denote the induced subgraph of distance $k$ by $\boldsymbol{G}_k$, and its corresponding adjacency matrix by $\mathbf{A}_{\boldsymbol{G}_k}$. To construct the normalized Laplacian matrix of $\mathbf{A}_{\boldsymbol{G}_k}$, we first need to compute the degree matrix as follows:

$$\mathbf{D}_{\boldsymbol{G}_k}(i,j) = \begin{cases} \sum_{k \in N_{\boldsymbol{G}_k}(i)} \mathbf{A}_{\boldsymbol{G}_k}(i,k) & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

Using the degree matrix $\mathbf{D}_{\boldsymbol{G}_k}$, we can define the normalized Laplacian matrix as follows:

$$\mathbf{L}_{\boldsymbol{G}_k} = \mathbf{D}_{\boldsymbol{G}_k}^{\frac{1}{2}}(\mathbf{I} - \mathbf{A}_{\boldsymbol{G}_k})\mathbf{D}_{\boldsymbol{G}_k}^{\frac{1}{2}} \tag{25}$$

where $\mathbf{I}$ is the identity matrix. The spectrum of matrix $\mathbf{L}_{\boldsymbol{G}_k}$, defined as the sorted list of its eigenvalues, contains critical information about the structural properties of the induced subgraph $\boldsymbol{G}_k$. However, for subgraphs of different radii this vector has varying sizes. To compare these eigenvalues, Patro *et al.* used a kernel density estimate of the eigenvalue distribution for the corresponding Laplacian matrix of different subgraphs (but with the same radii), which is then used as the topological signature of node $v_i$ represented by $\mathcal{S}_i^k$. To compute the distance between the topological signature of a pair vertices $v_i$ and $v_{i'}$, where $v_i \in \boldsymbol{G}$ and $v_{i'} \in \boldsymbol{H}$, they use a well-known information-theoretic distance [47] defined as follows:

$$d_k(i,i') = D_{JS}(\mathcal{S}_i^k, \mathcal{S}_{i'}^k). \tag{26}$$

In this formulation, $D_{JS}$ is the *Jensen-Shannon divergence*, which is a symmetrized version of *Kullback-Leibler divergence*.[1] They use a radial-basis kernel with standard deviation $\sigma^2$ and $\sigma = 10^{-2}$ for the density estimate. In order to combine distance measures defined over different subgraphs of varying degrees and compute the overall topological distance, they use the following definition:

$$\mathbf{D}_{topo}(i,i') = \sum_{k=1}^{4} d_k(i,i'). \tag{27}$$

In order to transform this these topological distances into topological similarity measure, we tried both min/max normalization, where $\mathbf{S}_{topo} = \frac{\mathbf{D}_{topo} - \mathbf{min}(\mathbf{D}_{topo})}{\mathbf{max}(\mathbf{D}_{topo}) - \mathbf{min}(\mathbf{D}_{topo})}$, as well as log-normalization, in which $\mathbf{S}_{topo} = -\mathbf{log}(\mathbf{D}_{topo})$. We evaluated both of these transformations over the NAPAbench dataset and observed that log-normalization yields better results. Thus, we used this transformation throughout our study. Finally, in order to mix sequence and topological similarities, we first scale each matrix so that the $n^{th}$ largest

1. Given a pair of distributions, $P$ and $Q$, their Kullback-Leibler divergence is defined as $D_{KL}(P \parallel Q) = \sum_i P(i)\frac{P(i)}{Q(i)}$. The KL divergence is not symmetric and the Jensen-Shannon divergence computes a convex combination of KL divergences between $P$ and $Q$ to the average of these two distributions: $D_{JS} = \frac{1}{2}D_{KL}(P \parallel M) + \frac{1}{2}D_{KL}(Q \parallel M)$, where $M = \frac{1}{2}(P+Q)$.

element of these matrices, where $n = \mathbf{min}(|(V)_G|, |(V)_G|)$, are equal. Then, we compute the convex combination as:

$$\mathbf{X}_{\text{GHOST-Topo}}(i, i') = \alpha \mathbf{S}_{\text{topo}}(i, i') + (1 - \alpha)\mathbf{W}(i, i') \quad (28)$$

where $\mathbf{W} = \mathbf{unvec}(\boldsymbol{w})$ represent the sequence similarity scores.

## ACKNOWLEDGMENTS

This work is supported by the Center for Science of Information (CSoI), an NSF Science and Technology Center, under grant agreement CCF-0939370, as well as by NSF Grant BIO-1124962, NSF Grant CCF-1149756, NSF Grant IIS-1422918, and the DARPA SIMPLEX Program. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE–AC04–94AL85000.

## REFERENCES

[1] R. Milo, S. Shen-Orr, S. Itzkovitz, *et al.*, "Network motifs: simple building blocks of complex networks.," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[2] N. Kashtan, S. Itzkovitz, *et al.*, "Efficient sampling algorithm for estimating sub-graph concentrations and detecting network motifs," *Bioinformatics*, vol. 20, pp. 1746–1758, 2004.

[3] V. Batagelj and A. Mrvar, "Pajek-analysis and visualization of large networks," *Springer-Verlag*, vol. 2265, pp. 77–103, 2003.

[4] F. Schreiber and H. Schwöbbermeyer, "Mavisto: a tool for the exploration of network motifs," *Bioinformatics*, vol. 21, pp. 3572–3574, 2005.

[5] S. Wernicke and F. Rasche, "Fanmod: a tool for fast network motif detection," *Bioinformatics*, vol. 22, pp. 1152–1153, 2006.

[6] Z. R. M. Kashani, H. Ahrabian, E. Elahi, *et al.*, "Kavosh: a new algorithm for finding network motifs.," *BMC bioinformatics*, vol. 10, no. 318, Jan. 2009.

[7] S. Mangan and U. Alon, "Structure and function of the feed-forward loop network motif," *Proceedings of the National Academy of Sciences*, vol. 100, no. 21, pp. 11 980–11 985, Oct. 2003.

[8] S. S. Shen-Orr, R. Milo, S. Mangan, *et al.*, "Network motifs in the transcriptional regulation network of Escherichia coli.," *Nature genetics*, vol. 31, no. 1, pp. 64–68, 2002.

[9] S. S. Chung, A. Pandini, A. Annibale, *et al.*, "Bridging topological and functional information in protein interaction networks by short loops profiling," *Scientific Reports*, vol. 5, p. 8540, Feb. 2015.

[10] A.-L. Barabási and Z. N. Oltvai, "Network biology: understanding the cell's functional organization," *Nature Reviews Genetics*, vol. 5, no. 2, pp. 101–113, Feb. 2004.

[11] S Wuchty, Z. N. Oltvai, and A.-L. Barabási, "Evolutionary conservation of motif constituents in the yeast protein interaction network," *Nature Genetics*, vol. 35, no. 2, pp. 176–179, Oct. 2003.

[12] L. H. Hartwell, J. J. Hopfield, S. Leibler, *et al.*, "From molecular to modular cell biology," *Nature*, vol. 402, no. 6761 Suppl, 1999.

[13] T. G. Kolda and J. R. Mayo, "Shifted power method for computing tensor eigenpairs," *SIAM J. Matrix Analysis Applications*, vol. 32, no. 4, pp. 1095–1124, 2011.

[14] O. Šváb, "Exploiting patterns in ontology mapping," in *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, K. Aberer, K.-S. Choi, N. Noy, *et al.*, Eds., ser. LNCS, vol. 4825, Berlin, Heidelberg: Springer Verlag, 2007, pp. 950–954.

[15] M. Chertok and Y. Keller, "Efficient high order matching," in *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2010, pp. 2205–2215.

[16] S. M. E. Sahraeian and B.-J. Yoon, *A Network Synthesis Model for Generating Protein Interaction Network Families*, 2012.

[17] L.-H. Lim, "Singular Values and Eigenvalues of Tensors: A Variational Approach," in *CAMSAP'05: Proceeding of the IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2005, pp. 129–132.

[18] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[19] R. B. Kelley and \textitet. el., "Conserved pathways within bacteria and yeast as revealed by global protein network alignment," *PNAS*, vol. 100(20), 2003.

[20] B. P. Kelley, B. Yuan, F. Lewitter, *et al.*, "PathBLAST: a tool for alignment of protein interaction networks," *Nucleic Acids Research*, vol. 32, no. Web-Server-Issue, pp. 83–88, 2004.

[21] R. Sharan, T. Ideker, B. P. Kelley, *et al.*, "Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data," *Journal of Computational Biology*, vol. 12, no. 6, pp. 835–846, 2005.

[22] R. Sharan, S. Suthram, R. M. Kelley, *et al.*, "Conserved patterns of protein interaction in multiple species.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 6, pp. 1974–1979, 2005.

[23] J Flannick, A Novak, B. S. Srinivasan, *et al.*, "Graemlin: general and robust alignment of multiple large interaction networks," *Genome Research*, vol. 16, no. 9, pp. 1169–1181, Sep. 2006.

[24] J. Flannick, A. F. Novak, C. B. Do, *et al.*, "Automatic parameter learning for multiple network alignment," in *RECOMB*, 2008, pp. 214–231.

[25] M. Koyutürk, A. Grama, and W. Szpankowski, "Pairwise local alignment of protein interaction networks guided by models of evolution," in *RECOMB*, 2005, pp. 48–65.

[26] M. Koyutürk, Y. Kim, U. Topkara, *et al.*, "Pairwise alignment of protein interaction networks," *Journal of Computational Biology*, vol. 13(2), pp. 182–199, 2006.

[27] R. Singh, J. Xu, and B. Berger, "Pairwise global alignment of protein interaction networks by matching neighborhood topology," in *Proceedings of the 11th Annual International Conference on Research in Computational Molecular Biology*, ser. RECOMB'07, Oakland, CA, USA: Springer-Verlag, 2007, pp. 16–31.

[28] R. Singh, J. Xu, and B. Berger, "Global alignment of multiple protein interaction networks with application to functional orthology detection," *PNAS*, vol. 105, no. 35, pp. 12 763–12 768, 2008.

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

[30] M. El-Kebir, J. Heringa, and G. W. Klau, "Lagrangian relaxation applied to sparse global network alignment," *CoRR*, vol. abs/1108.4358, 2011.

[31] M. Bayati, D. F. Gleich, A. Saberi, *et al.*, "Message-Passing Algorithms for Sparse Network Alignment," *ACM Trans. Knowl. Discov. Data*, vol. 7, no. 1, 3:1–3:31, Mar. 2013.

[32] O. Kuchaiev, T. Milenkovic, V. Memisevic, *et al.*, "Topological network alignment uncovers biological function and phylogeny.," *Journal of the Royal Society, Interface / the Royal Society*, vol. 7, no. 50, pp. 1341–54, Sep. 2010.

[33] T Milenković, W. L. Ng, W Hayes, *et al.*, "Optimal network alignment with graphlet degree vectors," *Cancer Inform*, vol. 9, 2010.

[34] O. Kuchaiev and N. Przulj, "Integrative network alignment reveals large regions of global network similarity in yeast and human.," *Bioinformatics (Oxford, England)*, vol. 27, no. 10, pp. 1390–6, May 2011.

[35] V. Memišević and N. Pržulj, "C-GRAAL: common-neighbors-based global GRAph ALignment of biological networks," *Integrative biology : quantitative biosciences from nano to macro*, vol. 4, no. 7, pp. 734–43, Jul. 2012.

[36] R. Patro and C. Kingsford, "Global network alignment using multiscale spectral signatures.," *Bioinformatics (Oxford, England)*, vol. 28, no. 23, pp. 3105–14, Dec. 2012.

[37] G. Ballard, T. G. Kolda, and T. Plantenga, "Efficiently computing tensor eigenvalues on a gpu," in *IPDPS Workshops*, 2011, pp. 1340–1348.

[38] W. K. Kim and E. M. Marcotte, "Age-Dependent Evolution of the Yeast Protein Interaction Network Suggests a Limited Role of Gene Duplication and Divergence," *PLoS Computational Biology*, vol. 4, no. 11, R. Nussinov, Ed., e1000232, Nov. 2008.

[39] J. C. Wootton and S. Federhen, "Statistics of local complexity in amino acid sequences and sequence databases," *Computers & Chemistry*, vol. 17, no. 2, pp. 149–163, Jun. 1993.

[40] W. R. Pearson and D. J. Lipman, "Improved tools for biological sequence analysis," *Proc. Natl. Acad. Sci.*, vol. 85, pp. 2444–2448+, 1988.

[41] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences.," *Journal of molecular biology*, vol. 147, no. 1, pp. 195–7, Mar. 1981.

[42] M. Bayati, M. Gerritsen, D. Gleich, *et al.*, "Algorithms for large, sparse network alignment problems," in *ICDM*, 2009, pp. 705–710.

[43] G. Kollias, M. Sathe, S. Mohammadi, *et al.*, "A fast approach to global alignment of protein-protein interaction networks.," *BMC research notes*, vol. 6, no. 1, p. 35, Jan. 2013.

[44] M Ashburner, C. A. Ball, J. A. Blake, *et al.*, "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium.," *Nature genetics*, vol. 25, no. 1, pp. 25–9, May 2000.

[45] C. Clark and J. Kalita, "A comparison of algorithms for the pairwise alignment of biological networks.," *Bioinformatics (Oxford, England)*, vol. 30, no. 16, pp. 2351–2359, 2014.

[46] T. G. Kolda and J. R. Mayo, "An adaptive shifted power method for computing generalized tensor eigenpairs," *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 4, pp. 1563–1581, 2014.

[47] A. Banerjee, "Structural distance and evolutionary relationship of networks," *BioSystems*, vol. 107, no. 3, pp. 186–196, 2012.