ANCA: Alignment-based Network Construction Algorithm

Kevin Chow^{1,†}, Ahmet Ay^{2,†}, Rasha Elhesha¹, Tamer Kahveci^{1,*}

¹ Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32608
{kchow23,relhesha,tkahveci}@ufl.edu

² Departments of Biology and Mathematics, Colgate University, Hamilton, NY 13346
aay@colgate.edu

* Corresponding author, † Equal contribution

ABSTRACT

Dynamic biological networks model changes in the network topology over time. However, often the topologies of these networks are not available at specific time points. Existing algorithms for studying dynamic networks often ignore this problem and focus only on the time points at which experimental data is available. In this paper, we develop a novel alignment based network construction algorithm, *ANCA*, that constructs the dynamic networks at the missing time points by exploiting the information from a reference dynamic network. Our experiments on synthetic and real networks demonstrate that *ANCA* predicts the missing target networks accurately, and scales to large biological networks in practical time. Our analysis of an *E. coli* protein-protein interaction network shows that *ANCA* successfully identifies key temporal changes in the biological networks. Our analysis also suggests that by focusing on the topological differences in the network, our method can be used to find important genes and temporal functional changes in the biological networks.

ACM Reference Format:

Kevin Chow^{1,†}, Ahmet Ay^{2,†}, Rasha Elhesha¹, Tamer Kahveci^{1,*}. 2018. ANCA: Alignment-based Network Construction Algorithm. In *Proceedings of 9th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (ACM-BCB'18)*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3233547.3233576>

1 INTRODUCTION

Biological networks describe the interactions between molecules such as genes and proteins [3]. These networks are involved in many cellular functions including cell differentiation and metabolism [9, 16]. Analysis of biological networks is thus the key to understand how living organisms function. Constructing the topology of biological networks from experimental data is the first step in studying biological networks.

Biological networks have dynamic topologies, i.e., over a period of time new interactions can appear or existing ones can disappear [1, 13]. For example, interactions between the genes which are involved in embryogenesis vary throughout development. These

dynamic changes in biological networks are key for correct development of the embryo. Major disorders such as cancer or external stimulants such as drugs are a few examples to many factors, which lead to dynamic alterations in network topology [8, 12].

Experimental techniques such as smFISH and ChIP-seq can provide a snapshot of a biological system's inner workings, but they are experimentally challenging and expensive [10, 14]. Thus, only a small fraction of possible molecular interactions and their temporal variations can be determined experimentally. Furthermore, experimental data is often not collected at specific time points due to either high cost or technical limitations. We say that a dynamic network has a missing time point if its topology is not known in at least one time point among a given sequence of time points and no experimental data is available as well at those time points. For example, in order to construct a dynamic network of a cancer patient, data from this patient should be collected at multiple time points. However, the patient may not be available at some of those times. One way to study evolving networks with such missing data is to restrict the analysis to the time points at which data is available. This results in missing key temporal changes. Constructing the topology of networks at such missing time points is of utmost importance to have a holistic view on how biological systems evolve.

Reconstructing dynamic biological networks from large-scale experimental data has been one of the major challenges in bioinformatics. Existing network construction methods often utilize gene expression levels, protein-protein interaction (PPI) assays and existing literature [5]. These approaches usually determine a static network that describes the molecular interactions at a specific time point at which the experimental data is available. Algorithms that construct dynamic networks exist, however, they also often rely on existence of experimental data at all time points, which may not be feasible due to the missing data [4]. Here, we approach the dynamic network construction problem from a different perspective. The central hypothesis of this paper is that given a target dynamic network with at least one missing time point, using a reference dynamic network of similar topology measured at the same and the missing time points, we can construct the target network at the missing time points.

Our contributions. In this study, we develop a novel method, *Alignment-based Network Construction Algorithm (ANCA)*, that constructs a dynamic network at the time points where data is not available in the presence of a reference network. Briefly, given a reference dynamic network with n time points, and a target dynamic network whose topology is not known in k of those n time points ($k < n - 1$); *ANCA* constructs the target network at k missing time

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM-BCB'18, August 29-September 1, 2018, Washington, DC, USA

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-5794-4/18/08...\$15.00

<https://doi.org/10.1145/3233547.3233576>

points by utilizing the alignment of the target and reference dynamic networks in the $n - k$ common time points. ANCA constructs the k missing networks iteratively. At each iteration, ANCA computes a joint alignment of the target and reference network pairs for all common time points at which the target network topology is available. Using this alignment, ANCA trains a classifier on all the aligned target and reference network edges that are observed at least once throughout the evolution of these networks. Using this classifier, it constructs the target network for one of the missing time points. In the following iterations, ANCA repeats this process iteratively to predict the target network in the next time point where data is not available until it completes construction of the target network at all missing time points. Note that the order in which we construct the target network in missing time points affects the accuracy of the prediction. We therefore repeat this in different orderings of time points and compute a consensus network topology. Our experimental results on real and synthetic datasets demonstrate that ANCA is accurate, robust and scalable. Our experiments on an *E. coli* PPI network suggest that our method can be used to discover key temporal changes in biological networks. Our results also suggest that ANCA will fill a gap in the dynamic network construction field, since all the existing algorithms—to our knowledge—ignore the time points that no data is available.

2 METHODS

In this section, we describe our method, *Alignment-based Network Construction Algorithm* (ANCA), that constructs a dynamic network's topology which is known only in a subset of a given set of time points in the presence of a reference dynamic network. In the following, first we describe the preliminary terms needed to describe ANCA and define the *Incomplete Network Construction (INC)* problem (Section 2.1). We then present our algorithm for constructing a dynamic network at one missing time point (Section 2.2.1). Finally, we describe our general algorithm for constructing dynamic networks at multiple missing time points (Section 2.2.2).

2.1 Preliminaries and problem definition

We denote a biological network as a graph $G = (V^G, E^G)$, where V^G and E^G represent the sets of vertices and edges, respectively. We represent a dynamic network as an ordered sequence of n networks; $\mathcal{G} = [G_1, G_2, \dots, G_n]$, where $G_i = (V^G, E_i^G)$ for $i = 1, 2, \dots, n$. Notice that, the vertices in the dynamic network are the same for all G_i , but the edges vary between consecutive networks.

Let $\mathcal{T} = [T_1, T_2, \dots, T_n]$ and $\mathcal{R} = [R_1, R_2, \dots, R_n]$ represent a target and a reference dynamic network, respectively. We denote the target and reference network at a specific time point i with $T_i = (V^T, E_i^T)$ and $R_i = (V^R, E_i^R)$, respectively. Dynamic network alignment seeks to find a function $\psi : V^R \rightarrow V^T \cup \{\emptyset\}$ that maps the vertices in V^R to those in V^T , by maximizing the similarity between the mapped target and reference vertices and their interaction topologies across all time points. Here mappings into $\emptyset$ indicate unaligned nodes. Notice that this is different than the classical pairwise network alignment as the final alignment function is invariant across different time points. We use the *Tempo* network alignment algorithm to construct the alignment function ψ [7].

We define the set of *aligned vertices* of the target network $\mathcal{T}$ as the vertices of V^T that have been mapped to the vertices of V^R , and denote them as V_a^T (i.e., $V_a^T = \{v \in V^T \mid \exists u \in V^R \ni \psi(u) = v\}$). Similarly, we define the set of *aligned edges* of T_i as the edges between the aligned vertices of T at the i th time point and denote it

with $E_{a,i}^T$. We call the induced subnetwork that is formed by the aligned vertices and edges of T_i as the *aligned target subnetwork at the i th time point* and denote it with $T_{a,i} = (V_a^T, E_{a,i}^T)$. We define the *aligned target dynamic subnetwork* as the ordered sequence of aligned target subnetworks $T_{a,i}$; $\mathcal{T}_a = [T_{a,1}, \dots, T_{a,n}]$. Similarly, we denote the aligned reference subnetworks at time point i as $R_{a,i} = (V_a^R, E_{a,i}^R)$ for $i = 1, \dots, n$, and the aligned reference dynamic network as $\mathcal{R}_a = [R_{a,1}, \dots, R_{a,n}]$. Next, we define the incomplete network construction problem.

DEFINITION 1. (INCOMPLETE NETWORK CONSTRUCTION (INC)). Consider a reference dynamic network, $\mathcal{R} = [R_1, \dots, R_n]$ where $R_i = (V^R, E_i^R)$ for $i = 1, \dots, n$. Let us define the set of q time points ($q < n$) at which the topology of a given target network is available along with the reference with $Q = \{t_1, t_2, \dots, t_q\}$. Here $Q \subset \{1, 2, \dots, n\}$ and $\forall i < q, t_i < t_{i+1}$. Let us denote the set of time points at which the target network topology is not known with $M = \{1, 2, \dots, n\} \setminus Q$. Let us define the target dynamic network with $\mathcal{T}_Q = [T_{t_1}, T_{t_2}, \dots, T_{t_q}]$, where $\forall t_i \in Q, T_{t_i} = (V^T, E_{t_i}^T)$. Let us denote the alignment of $\mathcal{T}$ with the reference dynamic network induced on time points Q (i.e., $[R_{t_1}, R_{t_2}, \dots, R_{t_q}]$) with $\psi : V^R \rightarrow V^T$. Given a dynamic network alignment function ψ , and target and reference networks $\mathcal{T}_Q$ and $\mathcal{R}$, INC problem seeks to construct the target network $\mathcal{T}$ at the $n - q$ missing time points (i.e., $\forall i \in M$, construct T_i).

To simplify the description of our algorithm, from now on, we assume that the size of the target dynamic network is same or larger than the reference dynamic network, i.e., $|V^T| \geq |V^R|$. That said, as we explain later, our solution does not depend on this assumption.

2.2 ANCA

In this section, we describe our algorithm to solve the INC problem. First, we describe our algorithm to construct the target dynamic network with one missing time point (Section 2.2.1). We then generalize our algorithm to multiple missing time points (Section 2.2.2).

2.2.1 Network construction at a single missing time point. Let $\mathcal{R} = [R_1, R_2, \dots, R_n]$ represent a reference dynamic network. Let $Q = \{t_1, t_2, \dots, t_{n-1}\}$ and $M = \{t_m\}$ represent time points within the set $\{1, 2, \dots, n\}$. Let $\mathcal{T}_Q = [T_{t_1}, T_{t_2}, \dots, T_{t_{n-1}}]$ denote a target dynamic network where its data is available at $n - 1$ time points, and missing at one time point t_m . Let $\mathcal{R}_Q = [R_{t_1}, R_{t_2}, \dots, R_{t_{n-1}}]$ represent the reference dynamic network induced on Q .

To construct T_{t_m} , first we align $\mathcal{T}_Q$ and $\mathcal{R}_Q$ using the *Tempo* algorithm [7]. Let $\psi : V^R \rightarrow V^T$ represent the alignment function with respect to *Tempo*. Recall that ψ defines a subnetwork of the target network at each time point t_i as aligned target subnetwork, denoted with T_{a,t_i} . We define the remaining target subnetwork at each t_i as complementary, and denote it with T_{c,t_i} . More specifically, the complementary network $T_{c,t_i} = (V_c^T, E_{c,t_i}^T)$ contains all nodes in V_c^T and all the remaining edges in $E_{t_i}^T$ (i.e., $E_{c,t_i}^T = E_{t_i}^T \setminus E_{a,t_i}^T$).

We construct the edges in the aligned target subnetwork T_{a,t_m} and the complementary target subnetwork T_{c,t_m} using two different approaches. Next, we describe these two approaches.

Construction of the aligned target subnetwork. Let us denote the set of edges present in the aligned reference dynamic subnetwork and existing target dynamic network which appear in at least one observed time point with E , i.e., $E = (\cup_{i=1}^{n-1} E_{a,t_i}^T) \cup (\cup_{j=1}^n E_{a,j}^R)$. We define internal time points among the sequence of time points in the given dynamic networks as the set of all time points in Q except the first and last time point, i.e., $\{t_2, t_3, \dots, t_{n-2}\}$.

We use a Support Vector Machine (SVM) classifier to predict whether each edge $e \in E$ exists at the aligned target subnetwork at the missing time point (i.e., T_{a,t_m}) [6].

We generate a feature vector for each edge $e \in E$ at each internal time point t_i using the topology of five networks, namely the reference and target networks at time points immediately before and after that time point as well as the reference network at that time point (i.e., $T_{t_{i-1}}, T_{t_i}, R_{t_{i-1}}, R_{t_i}$, and $R_{t_{i+1}}$).

First, we define the terms that are needed to explain the training features. Let u and v be two nodes in a network G , and let e be the edge that connects them, i.e., $e = (u, v)$. We define an indicator function denoted with $B(e, G)$. This function returns 1 if e exists in G , and 0 otherwise. We define the distance between u and v on G as the length of the shortest path between u and v on G . We define d th degree neighbors of u in G as the set of all nodes, which have distance d to node u , and denote it as $F_d(u, G)$. Similarly, we define the d th degree neighbors of edge e denoted with $N_d(e, G)$ as the union of all the d th degree neighbors of u and v , i.e., $N_d(e, G) = F_d(u, G) \cup F_d(v, G)$. Finally, consider two networks G_r and G_s in a dynamic network with $r < s$ and an edge $e \in G_r \cap G_s$. We define the number of d th degree neighbor insertions and removals from G_r to G_s as $N_{e,d}^i(G_r, G_s) = |N_d(e, G_s) \setminus N_d(e, G_r)|$ and $N_{e,d}^r(G_r, G_s) = |N_d(e, G_r) \setminus N_d(e, G_s)|$, respectively.

Consider an edge $e \in E$. We collect seventeen features for this edge at time point t_i as follows. The first five features indicate existence (1) / absence (0) of e in five aligned subnetworks, i.e., $B(e, T_{a,t_{i-1}})$, $B(e, T_{a,t_{i+1}})$, $B(e, R_{a,t_{i-1}})$, $B(e, R_{a,t_i})$ and $B(e, R_{a,t_{i+1}})$. Note that we skip the network T_{a,t_i} to mimic that T_{t_i} is missing. The remaining twelve features for edge e at the internal time point t_i represent the topological changes in the neighborhood of this edge in three pairs of networks $\{T_{t_{i-1}}, T_{t_{i+1}}\}$, $\{R_{t_{i-1}}, R_{t_i}\}$ and $\{R_{t_{i-1}}, R_{t_{i+1}}\}$. More specifically, for each network pair we record four values; first/second degree insertion/removals (e.g., $N_{e,1}^i(T_{t_{i-1}}, T_{t_{i+1}})$).

For a target dynamic network where data is available at $n - 1$ time points, we generate the training dataset by collecting the features information at $n - 3$ sequential time triplets where target network data is available (i.e., $\forall t_i \in [t_2, \dots, t_{n-2}]$ we use target and reference networks at the time points t_{i-1} , t_i and t_{i+1}). We generate a training sample at each t_i independently for each edge $e \in E$, and use these $|E| \times (n - 3)$ samples as the training set. Notice that an edge $e \in E$ may exist at some time point t_i but may be absent at another time point t_j . When e exists, we create a positive sample. When it is absent we construct a negative sample. We generate the test data using the triplets of the target and reference networks around the missing time point t_m (i.e. we use available target and reference networks at the time points $t_m - 1$, t_m and $t_m + 1$). Thus, we have $|E|$ test samples.

Construction of the complementary target subnetwork.

Notice that, unlike the aligned subnetwork, the complementary subnetwork has no topological evidence of absence/presence of its edges from the reference network. We therefore predict the edges in the complementary target subnetwork at the missing time point t_m (i.e., T_{c,t_m}) by considering the two complementary target subnetworks immediately before and after the missing time point (i.e., T_{c,t_m-1} and T_{c,t_m+1}). We use two conjectures to construct T_{c,t_m} ; (i) The edge insertion/removal is a rare event. (ii) Although different edges may have different mutation rates, the mutation rate of each edge remains unchanged through the evolution of the network.

Consider an edge e which exists either immediately before or after the missing time point in the complementary target network (i.e., $e \in E_{c,t_m-1}^T \cup E_{c,t_m+1}^T$). There are three possibilities; (i) e exists in both time points (i.e., $e \in E_{c,t_m-1}^T \cap E_{c,t_m+1}^T$), (ii) e is inserted (i.e., $e \in E_{c,t_m+1}^T \setminus E_{c,t_m-1}^T$), (iii) e is removed (i.e., $e \in E_{c,t_m-1}^T \setminus E_{c,t_m+1}^T$). If e is in the first category, following from the first conjecture above, we include e in the target network at the missing time point. If e is in the one of the last two categories (i.e., e is inserted or removed between $t_m - 1$ and $t_m + 1$) we need to determine if insertion/removal takes place before or after t_m . We use the second conjecture above to make that decision with a biased coin flip approach unique to e . We set the success probability of the biased coin to the fraction of complementary target subnetworks (i.e., $\mathcal{T}_{c,Q} = [T_{c,t_1}, T_{c,t_2}, \dots, T_{c,t_{n-1}}]$) which contain e . We randomly insert e into E_{c,t_m}^T with this probability. For example, if an edge between nodes 1 and 2 appears three out of four time points, we use $3/4$ as the success rate for this edge. Notice that our algorithm is nondeterministic. Hence, we repeat the coin flip step ten times, and report the average and standard error of the prediction accuracy.

Note that, if the target network is smaller than the reference network, we only predict the edges in the aligned region since there is no complementary target network.

2.2.2 Network construction at multiple missing time points. Here, we generalize ANCA to construct a dynamic network with missing data at multiple consecutive time points. In the following, we first describe how we construct the target network at all missing time points one time point at a time for a specific ordering of time points. We then explain how we extend it to construct the target network independent of the order of prediction.

Prediction of a particular ordering. Let $\mathcal{R} = [R_1, R_2, \dots, R_n]$ and $\mathcal{T}_Q = [T_{t_1}, T_{t_2}, \dots, T_{t_{n-k}}]$ represent all the existing reference and target dynamic networks. Let $\mathcal{R}_Q$ represent the reference dynamic network induced on the time points where target dynamic network data is available, i.e., $Q = \{t_1, t_2, \dots, t_{n-k}\}$. Let $\mathcal{T}_M = [T_{m_1}, T_{m_2}, \dots, T_{m_k}]$ represent the k consecutive missing target networks, where $M = \{1, 2, \dots, n\} \setminus Q$. We construct all missing target networks using the following iterative procedure. In the first iteration, we ignore the last $k - 1$ missing time points and construct the first missing target network T_{m_1} as described above using the target dynamic network $\mathcal{T}_Q$ and reference dynamic network $\mathcal{R}_{Q \cup \{m_1\}}$ (see Section 2.2.1). Here $\mathcal{R}_{Q \cup \{m_1\}}$ represents the reference dynamic network induced on the time points $Q \cup \{m_1\}$. Next, we insert T_{m_1} into the existing target dynamic network $\mathcal{T}_Q$. Thus, the number of missing time points in the target dynamic network $\mathcal{T}_{Q \cup \{m_1\}}$ reduces by one. Now that the target dynamic network has one more time point, we align the target and reference networks again at those $n - k + 1$ time points. Notice that the new alignment may yield a different mapping function between the reference and target networks, leading to different aligned and complementary networks. We generate the training and testing data again using the new alignment and retrain the SVM classifier using this data. In the second iteration, we similarly use the reference dynamic network $\mathcal{R}_{Q \cup \{m_1, m_2\}}$ and the target dynamic network $\mathcal{T}_{Q \cup \{m_1\}}$ to construct the missing target network T_{m_2} . We continue this iterative process until we construct all k missing target networks.

Network construction order. In the above iterative procedure, we have constructed the missing target networks in a certain order. However, construction order of the missing target networks ($\mathcal{T}_M$)

affects their accuracy since each missing network's construction rely on the networks that are previously predicted. In order to eliminate the bias based on construction order, we construct the missing target networks in all possible $k!$ orderings, and then combine their results. Notice that as the number of missing time points gets larger the number of experiments increases dramatically. For this reason, a sampling approach can be employed if the number of missing networks is large. Next, we explain how we combine the $k!$ network predictions.

Consensus. In the above iterative step, we construct each edge of a missing target $(k - 1)!$ times in each of the iterations $\{1, 2, \dots, k\}$. For example, for a target dynamic network with missing data at the time points 7, 8 and 9, we predict each edge e of target network T_7 twice in the first iteration $((7, 8, 9)$ and $(7, 9, 8))$, twice in the second iteration $((8, 7, 9)$ and $(9, 7, 8))$ and twice in the third iteration $((8, 9, 7)$ and $(9, 8, 7))$. We conjecture that the accuracy of our predictions for a missing target network gets better as the iteration at which we construct that network gets later. This is because as the iterations increase, the size of the training set increases linearly as well. We therefore use a weighted voting scheme based on the $k!$ predictions of an edge to determine whether this edge should exist in the missing target network. We consider the later time points with larger weights (i.e., more decisive), more specifically the prediction on whether an edge should exist or not at a given time point at the j th iteration ($1 \leq j \leq k$) has weight k^{j-1} . To predict the existence of an edge in a missing target network, we accumulate the votes for and against its existence across all $k!$ orderings and pick the winner. If the votes are identical, we use a fair coin flip approach to determine this edge's existence in the missing target network at the specific time point.

3 RESULTS

In this section, we experimentally evaluate our method on real and synthetic networks. We first describe the datasets and accuracy metrics used in our experiments. We then present our results.

REAL DATASET. We analyze the dynamic *E. coli* PPI network under varying environmental conditions. We use the PPI network data from the String database with interaction confidence greater than or equal to 900 [15], and the *E. coli* gene expression dataset, *GSE20305*, from the GEO database [11]. The expression dataset contains time resolved response of *E. coli* under five different conditions (glucose-lactose diauxic shift, cold, heat and oxidative stress, and control). Control, and cold and hot stress conditions contain data at eight time points. Oxidative stress and glucose-lactose diauxic shift conditions have data at twelve and six time points, respectively.

SYNTHETIC DATASET. We generate random target and reference dynamic networks to test our algorithm under controlled settings using four parameters: network size (i.e., number of nodes), number of time points, and temporal and cold evolution rates. We generate the target network at the first time point (T_1) using the Barabasi-Albert model [2] as this model is commonly used in the literature to simulate real networks. We fix the average node degree to four, and vary the network size to take values from $\{100, 250, 500\}$. We vary the number of time points as 5, 10, and 20. The last two parameters above control how much the network topology changes from one time point to the next. Briefly we generate the initial target network T_1 randomly. We then select a random connected subnetwork R_1 of this target network as the reference network. We use reference network size of 50 in all experiments. We label all nodes in T_1 and R_1 as **cold** and the remaining nodes as **hot**. We say that an edge is

cold (i.e., slow evolving) if it connects two cold nodes, otherwise we say that it is hot (i.e., fast evolving). Let us denote the cold, hot, and temporal evolution rates as ϵ^c , ϵ^h and ϵ ($\epsilon^h \geq \epsilon^c$) respectively with $\epsilon^h = (\epsilon - \epsilon^c\gamma)/(1 - \gamma)$ where γ is the ratio of cold edges to the total number of edges in the target network. We generate the target and reference networks T_i and R_i at the i th ($i > 1$) time point as follows. To generate R_i from R_{i-1} , we iterate over all the edges of R_{i-1} . We remove each edge with probability ϵ^c , and reinsert it between two random nodes. We evolve target network similarly. We use probabilities ϵ^c and ϵ^h to remove and reinsert cold and hot edges in target network, respectively. In this study, we choose ϵ and ϵ^c to take values from $\{0.1, 0.2, 0.4, 0.8\}$ and $\{0.05, 0.1, 0.2\}$, respectively. We generate pairwise homology score between target and reference nodes using log-normal distribution with mean μ and standard deviation σ . We use $\mu = 2$ and $\mu = 4$ for cold and hot nodes, respectively. We use $\sigma = 0.25$ for both node types.

NETWORK CONSTRUCTION ACCURACY METRICS. We measure accuracy of our method on both the aligned target subnetwork and the entire target network at the time points where data is missing. Let P , TP and FP represent number of positive, true positive and false positive samples, respectively. Similarly, let N , TN and FN denote number of negative, true negative and false negative samples, respectively. We separate our evaluation metrics for the aligned region and the overall network since we use different methods to predict the edges in the aligned target subnetwork and in its complement. We use *balanced accuracy* (BAC), which is defined as $(TP/P + TN/N)/2$, to evaluate the accuracy of our method on the aligned target subnetwork. We choose this metric due to its less biased evaluation of imbalanced class data. We use *F1-score* to evaluate the accuracy of our algorithm on the entire target network. *F1-score* is equal to $2 \times TP / (2 \times TP + FP + FN)$.

3.1 Evaluation on synthetic networks

In this section, we evaluate ANCA on synthetic dataset. We focus on the effects of five parameters; network size, number of time points, number of missing time points, and temporal and cold rates. In our experiments, we choose network size, number of time points, temporal rate, cold rate, and number of missing time points to take values from $\{100, \underline{250}, 500\}$, $\{5, \underline{10}, 20\}$, $\{0.1, 0.2, \underline{0.4}, 0.8\}$, $\{0.05, \underline{0.1}, 0.2\}$ and $\{\underline{1}, 2, 3\}$, respectively. In each experiment, we fix four parameters to the bold underlined value, and vary the fifth one. We fix the reference network size to 50. To the best of our knowledge, ANCA is the first method to solve the INC problem using reference network. We thus compare our method to a biased coin flip based random network construction approach. This approach assigns a probability of success to each target edge as the fraction of the time points at which it is observed. It then iterates over all edges, and includes each edge with the probability of that edge at the missing time point. We repeat the random coin flip 10 times and compute the mean and standard deviation of the accuracy. For all our synthetic experiments running time varies between 54 and 915 seconds, which suggests that ANCA is feasible for large biological networks. The number of missing time points affects the running time most since for k missing networks we perform $k!$ experiments. For large k , as mentioned above, a sampling approach can be utilized to keep the computational cost at a feasible level.

Network construction accuracy on synthetic networks.

Here, we evaluate the accuracy of our method on the target network and the aligned subnetwork at the missing time points using *F1-score* and *BAC*, respectively. Our results demonstrate that ANCA is

Figure 1: BAC and F_1 -score comparison under varying parameters; (a) target size, (b) number of time points, (c) temporal rate, (d) cold rate, and (e) number of missing time points. Gray bars and white bars represent the accuracy levels of ANCA and biased coin flip, respectively. Vertical error bars represent two standard errors from the mean.

significantly more accurate than the biased coin flip approach for all parameter settings (Figure 1). In the following subsections, we focus on the effects of each tested parameter.

Effects of network size (Figures 1a and 1f). We observe that our method’s balanced accuracy increases slightly as the target network size increases (Figure 1a). Increase in BAC can be attributed to decreased hot evolution rate for increasing network size when both cold rate and temporal rate is fixed. This leads to dynamic networks with less edge variation among the hot nodes of the target network, and thus higher quality training data for ANCA. Note that, ANCA’s accuracy on the entire target network does not change as the network size increases (Figure 1f). As mentioned before, as the target network size increases hot rate decreases. However, the number of hot edges we predict using the less accurate biased coin flip increases as well. As a result, although we get better accuracy for the aligned subnetwork, the overall accuracy on the target network remains equally high.

Effects of the number of time points (Figures 1b and 1g). We observe that as the number of time points increases, balanced accuracy for our method grows rapidly to almost 100% (Figure 1b). Accuracy improvement is likely associated with increase in the training dataset size; as the number of time points increases we get more network edges that could be used for training ANCA. Accuracy of ANCA on the entire target network does not change significantly as the number of time points increases (Figure 1g). This could be explained as follows. Since we generate the dynamic networks using perturbation, the number of edges that should be predicted using a biased coin flip increases as the number of time points increase. Notice that accuracy of the biased coin flip approach (white bars) significantly decreases with growing number of time points. Recall that ANCA uses biased coin flip to predict the presence/absence of a small number of edges (edges in the symmetric difference of two consecutive complementary subnetworks). Therefore, ANCA’s accuracy in the complementary subnetwork is not affected significantly. In summary, although ANCA’s accuracy increases in the aligned subnetwork and decreases slightly in the complementary subnetwork, its accuracy on the entire target network remains high.

Effects of temporal rate (Figures 1c and 1h). Our results demonstrate that as temporal rate increases, ANCA’s accuracy on the aligned subnetwork decreases slightly, yet it remains high even for large temporal evolution rates. On the other hand, ANCA’s accuracy on the entire target network decreases significantly. The coin flip approach shows a similar trend. The reduction in accuracy can be

explained by the decrease in the data quality. As the temporal rate increases, the overall network topology changes at larger quantities. For instance when $\epsilon = 0.8$, the network topologies of subsequent networks vary by 80% on the average. Such large variation results in more edges that should be predicted using ANCA and biased coin flip. Thus, although the alignment between target and reference networks gets better due to significant divergence between cold and hot edges, our method’s accuracy decreases.

Effects of cold rate (Figures 1d and 1i). We observe that as cold rate increases, ANCA’s accuracy on the aligned subnetwork decreases slightly, but its accuracy on the entire target network does not change. The decrease in the BAC is expected since our training data quality diminishes with growing cold rate. We get more edge variation in the aligned target networks at different time points. Recall that, as cold rate increases, hot rate decreases in our target networks since temporal rate is fixed in our synthetic networks. Thus, as cold rate increases, ANCA’s accuracy in aligned subnetwork decreases, and its accuracy in the complementary network increases. Thus, we get comparable accuracy on the entire network as cold rate increases.

Effects of number of missing time points (Figures 1e and 1j). Our method’s accuracy is negatively affected by the number of missing time points. This could be explained by reduction in dataset quality due to increase in number of missing time points. Decrease in data quality leads to poor alignment of target and reference networks, and development of an inaccurate edge classifier. However, the decrease in accuracy is not substantial. This suggests that ANCA could be practically used for dynamic network construction for real biological networks even multiple time points are missing.

3.2 Evaluation on real networks

To show applicability of ANCA on real biological networks, here, we apply it to a dynamic *E. coli* PPI network under control/normal (*N*) condition, and four environmental perturbations; glucose-lactose diauxic shift (*L*), cold (*C*), heat (*H*) and oxidative (*O*) stress. Dynamic networks *N*, *C*, *H*, *L* and *O* have 375, 663, 2445, 3204 and 3183 nodes, respectively. Dynamic networks *C*, *H* and *N* have data at eight time points (1–8). Dynamic networks *L* and *O* have data at twelve (1–12) and six (2–7) time points, respectively. In each computational experiment, we choose our target and reference dynamic networks from these five PPI networks. In the cases that the number of time points for the target and the reference dynamic networks mismatch, we use the data at the common time points. For each experiment, we remove the target network data at one of the four common

Figure 2: Accuracy of ANCA method on *E. coli* PPI network under five experimental conditions. BAC (a), F_1 -score (b) show the accuracy of the ANCA method on average for aligned and complementary target subnetworks, respectively.

internal time points (i.e., 3, 4, 5 or 6). We use ANCA to construct this removed target network using all five dynamic *E. coli* networks as our reference. Figure 2 presents accuracy of our method for constructing a missing target network under all reference networks. For the diagonal entries we use the same dynamic network as both target and reference, and assume that each protein in the target network is aligned to itself in the reference network. Notice that our algorithm achieves close to 100% accuracy for this case. This suggests that given the right reference network and alignment function ANCA could achieve perfect accuracy.

Network construction accuracy on real networks. In all our experiments, we observe that ANCA is highly accurate on constructing the missing target network. For all target and reference network choices, ANCA achieves BAC and F_1 -score over 80% (Figure 2). This suggests that for the *E. coli* PPI network, dynamic networks from different experimental conditions can be used as reference for each other. Biological data collection is often either costly or technically challenging under some experimental conditions. Thus, we hypothesize that ANCA will be very helpful to construct temporal changes in biological networks under these experimental conditions.

Effects of target and reference network choice. Our analysis suggests that ANCA's accuracy depends on both the choice of target and reference network. In general, constructing the aligned target subnetwork of *L* and *H* is easier than that of *C*, *N* and *O* (Figure 2a). Similarly, on average, it is easier to construct the entire target dynamic network for *L* and *N* than for *C*, *H* and *O* (Figure 2b). Our analysis also shows that the dynamic networks *C* and *N* serve as better reference networks than the dynamic networks *H*, *L* and *O*.

Our method shows asymmetric behavior if the target and reference network change roles, i.e., target network is used as reference, and reference network is used as target. For example, although dynamic network *N* serves as a very good reference network for constructing the aligned subnetwork of *O*, using *O* as reference does not yield good accuracy for constructing the aligned subnetwork of *N* (Figure 2a). We believe that this discrepancy arises from the differences in the dynamic network alignment. The *Tempo* method provides slightly different alignments if target and reference networks change roles. Thus, ANCA trains on different data, which leads to slightly different network construction accuracy levels.

Effects of reference network size. Our analysis demonstrate that, in general, smaller networks serve as better reference networks. As the reference network size gets larger from left to right, ANCA's accuracy on the aligned target subnetwork gets slightly better for the dynamic networks *C* and *N* (Figure 2a). However, accuracy

on aligned target subnetwork for the dynamic networks *H*, *L* and *O* gets much worse. Similarly, as the reference network size gets larger, the F_1 -score for our method does not change much for the dynamic networks *C*, *L*, *H* and *N*. However, it gets worse for the dynamic network *O* (Figure 2b). We conjecture that this is because *Tempo* generates better alignment between target and reference networks for small reference network sizes. This results in better training data for ANCA, and thus, higher accuracy on the aligned target subnetwork. Since the size of the complementary target subnetwork increases as the reference network size decreases, we use the less accurate coin flip on more edges. This leads to less accurate predictions on the complementary target subnetwork. Thus, our overall accuracy in general stays similar on the entire target network.

Effects of number of time points. Our analysis shows that there is no association between ANCA's accuracy and the number of time points that reference network data is available. This is in contrast to our results on synthetic networks; as the number of time points gets larger ANCA becomes more accurate. Note that in synthetic networks while we increase the number of time points we fix all the other network parameters. This is not the case in real networks since we do not have control on the other network parameters. For example, we observe that dynamic networks with eight time points (*C* and *N*) serve as better reference networks than the dynamic networks with six (*L*) and twelve (*O*) time points (Figure 2a). This observation suggests that the dynamic network under oxidative stress with twelve time points has potentially different network topology than the other four dynamic networks.

REFERENCES

- [1] Ziv Bar-Joseph and et al. 2012. Studying and modelling dynamic biological processes using time-series gene expression data. *Nature Reviews Genetics* 13, 8 (2012), 552.
- [2] Albert-Laszlo Barabasi and Reka Albert. 1999. Emergence of scaling in random networks. *Science* 286, 5439 (1999), 509–512.
- [3] Albert-Laszlo Barabasi and Zoltan N Oltvai. 2004. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics* 5, 2 (2004), 101.
- [4] Allister Bernard and Alexander J Hartemink. 2005. Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data. In *Bioinformatics 2005*. World Scientific, 459–470.
- [5] Lian En Chai and et al. 2014. A review on the computational approaches for gene regulatory network construction. *Computers in Biology and Medicine* 48 (2014), 55–65.
- [6] Nello Cristianini and et al. 2000. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press.
- [7] Rasha Elhesha and et al. 2018. Identification of co-evolving temporal networks. *bioRxiv* (2018). <https://doi.org/10.1101/303974>
- [8] Jennifer Ann Hadley and et al. 2016. Change in brain network topology as a function of treatment response in schizophrenia: a longitudinal resting-state fMRI study using graph theory. *NPJ Schizophrenia* 2 (2016), 16014.
- [9] Trey Ideker and et al. 2001. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science* 292, 5518 (2001), 929–934.
- [10] Ni Ji and Alexander Van Oudenaarden. 2005. Single molecule fluorescent in situ hybridization (smFISH) of *C. elegans* worms and embryos. *WormBook* (2005).
- [11] Szymon Jozefczuk and et al. 2010. Metabolomic and transcriptomic stress response of *Escherichia coli*. *Molecular Systems Biology* 6, 1 (2010), 364.
- [12] Mark DM Leiserson and et al. 2015. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nature Genetics* 47, 2 (2015), 106.
- [13] Shane Neph and et al. 2012. Circuitry and dynamics of human transcription factor regulatory networks. *Cell* 150, 6 (2012), 1274–1286.
- [14] Peter J Park. 2009. ChIP-seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics* 10, 10 (2009), 669.
- [15] Damian Szklarczyk and et al. 2016. The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. *Nucleic Acids Research* 45 (2016), 362–368.
- [16] Nir Yosef and et al. 2013. Dynamic regulatory network controlling TH17 cell differentiation. *Nature* 496, 7446 (2013), 461.