Supplementary material: Aligning dynamic networks with DynaWAVE

Vipin Vijayan and Tijana Milenković
Department of Computer Science and Engineering, ECK Institute for Global Health,
Interdisciplinary Center for Network Science and Application (iCeNSA),
University of Notre Dame, IN 46556. E-mail: tmilenko@nd.edu

S1 Methods

S1.1 WAVE

WAVE greedily maximizes $\beta S_E + (1-\beta)S_N$, where S_E is the WEC edge conservation measure, and $S_N = \frac{1}{|V_1|}\sum_{u\in V_1} s(u,u')$ is the node conservation measure with $s(u,u')$ being the node similarity between $u \in V_1$ and $u' \in V_2$, and β is a parameter between 0 and 1 that balances between the two conservation types. Node similarities $s(\cdot,\cdot)$ can be based on network topology only (e.g., graphlet-based node similarities), information external to network topology such as protein sequence similarities, or a combination of the two. Two node similarities, $s_1(u,v)$ and $s_2(u,v)$, between nodes u and v can be combined, for example, using the convex combination $\alpha s_1(u,v) + (1-\alpha)s_2(u,v)$. Specifically in this paper, we use graphlet-based node similarities (Milenković and Pržulj, 2008) and set the β parameter to 0.5. Graphlets, for static networks, are small, connected, induced sub-graphs of a larger static network. Graphlets can be used to describe the local topology of a node in a static network. We use the graphlet-based node similarity $s(\cdot,\cdot)$ for static networks as described in Vijayan *et al.* (2017).

S1.1.1 WEC

We continue our discussion of WEC from Section 2 of the main paper. Similar to MAGNA++'s S^3, WAVE's WEC also counts the number of conserved edges, but unlike S^3 that treats each conserved edge the same, WEC favors conserved edges with similar end-nodes over conserved edges with dissimilar end-nodes. That is, for each edge $(u,v) \in E_1$ aligned to edge $(u',v') \in E_2$, the conserved edge is weighted by $s(u,u')$ and $s(v,v')$, the node similarity between nodes $(u,u') \in V_1 \times V_2$ and nodes $(v,v') \in V_1 \times V_2$, respectively.

Formally, given two static networks $G_1(V_1,E_1)$ and $G_2(V_2,E_2)$, assuming without loss of generality that $|V_1| \leq |V_2|$, and a static NA $f : V_1 \rightarrow V_2$ between them,

$$\text{WEC} = \sum_{(u,v)\in V_1\times V_1} \frac{s(v,v')\,\mathbb{1}\left[(u,v)\in E_1 \wedge (u',v')\in E_2\right]}{2\min(|E_1|,|E_2|)},$$

where $U \times V$ is the Cartesian product of sets U and V, $u' = f(u)$, $v' = f(v)$, and $c(u,v) = 1$ if the edge $(u,v) \in E_1$ is conserved and $c(u,v) = 0$ otherwise. That is, $c(u,v) = \mathbb{1}\left[(u,v)\in E_1 \wedge (u',v')\in E_2\right]$, where $\mathbb{1}[p] = 1$ if p is true and $\mathbb{1}[p] = 0$ if p is false.

S1.1.2 WAVE alignment strategy

We continue our discussion of WAVE's alignment strategy from Section 2 of the main paper. Above, we described WEC when given two static networks $G_1(V_1,E_1)$ and $G_2(V_2,E_2)$, and a static NA $f : V_1 \rightarrow V_2$ between them. That is, f describes a mapping from V_1 to V_2 for all nodes $v \in V_1$. However, it is also possible to describe WEC when given a partial mapping from V_1 to V_2, i.e., when given a partial mapping $f^* : V_1^* \rightarrow V_2$ where $V_1^* \subset V_2$. In order to motivate the WAVE alignment strategy, we will now describe WEC when given a partial mapping between two static networks. Given a partial mapping $f^t : V_1^t \rightarrow V_2$, where only the nodes in $V_1^t \subset V_1$ are aligned, we are able to calculate the partial WEC:

$$\text{WEC}^t = \sum_{(u,v)\in V_1^t\times V_1^t} \frac{s(v,v')\,\mathbb{1}\left[(u,v)\in E_1 \wedge (u',v')\in E_2\right]}{2\min(|E_1|,|E_2|)}.$$

Now, what happens to WEC^t when we align a new node $w \in V_1$ to the partial mapping? What is the marginal improvement in WEC^t when we add a new node $w \in V_1$ that is mapped to node $w' \in V_2$? Formally, let $f^{t+1} : V_1^{t+1} \rightarrow V_2$ be the partial

mapping when we add to V_1^t a new node $w \in V_1$ that maps to node $w' \in V_2$; that is, $V_1^{t+1} = \{w\} \cup V_1^t$ and $f^{t+1}(V^t) = \{w'\} \cup f^t(V^t)$ where $w' = f^{t+1}(w)$. Then, the marginal improvement of WEC^t when adding the aligned node pair (w, w') to f^t is:

$$\begin{aligned}
\text{WEC}^{t+1} - \text{WEC}^t &= \sum_{(u,v) \in V_1^{t+1} \times V_1^{t+1}} \frac{s(v,v')\,\mathbb{1}\left[(u,v) \in E_1 \wedge (u',v') \in E_2\right]}{2\min(|E_1|,|E_2|)} - \sum_{(u,v) \in V_1^{t} \times V_1^{t}} \frac{s(v,v')\,\mathbb{1}\left[(u,v) \in E_1 \wedge (u',v') \in E_2\right]}{2\min(|E_1|,|E_2|)} \\
&= \sum_{u \in V_1^t} \frac{s(w,w')\,\mathbb{1}\left[(u,w) \in E_1 \wedge (u',w') \in E_2\right]}{2\min(|E_1|,|E_2|)} + \sum_{v \in V_1^t} \frac{s(v,v')\,\mathbb{1}\left[(w,v) \in E_1 \wedge (w',v') \in E_2\right]}{2\min(|E_1|,|E_2|)} \\
&= \sum_{u \in V_1^t} \frac{(s(w,w') + s(u,u'))\,\mathbb{1}\left[(u,w) \in E_1 \wedge (u',w') \in E_2\right]}{2\min(|E_1|,|E_2|)} \qquad \text{(since we have undirected edges)} \\
&= \sum_{u \in V_1^t \wedge (u,u') \in N_w \times N_{w'}} \frac{s(w,w') + s(u,u')}{2\min(|E_1|,|E_2|)},
\end{aligned}$$

where N_u contains the neighbors of node u. So, the marginal improvement of WEC^t weigh each conserved edge using the node similarities of w and its neighbors N_w, as well as the node similarities of w' and its neighbors $N_{w'}$.

This suggests a greedy alignment strategy: given a partial alignment f^t, we may choose a new aligned pair (w, w') such that $\text{WEC}^{t+1} - \text{WEC}^t$ is maximized; furthermore, due to the above equation, we can see that this maximization procedure will depend only on the nodes that are already aligned and their neighbors. We then repeat this maximization process until all nodes in V_1 are aligned. We describe this alignment strategy below.

Note that as explained in Section 2 of the main paper, WAVE maximizes $\beta S_E + (1-\beta) S_N$, where S_E is the WEC edge conservation measure, and $S_N = \frac{1}{|V_1|} \sum_{u \in V_1} s(u,u')$ is the node conservation measure with $s(u,u')$ being the node similarity between $u \in V_1$ and $u' \in V_2$. Thus, the goal of WAVE is to maximize the alignment quality measure $AQ = \beta WEC + (1-\beta) S_N$. We now describe WAVE's alignment strategy.

Given an initial empty mapping $f^0 : V_1^0 \to V_2$, where $V_1^0 = \varnothing$, the associated alignment quality score $\text{AQ}^0 = 0$. Then, the aligned node pair with the highest marginal increase in AQ is the node pair (u,u') with the highest node similarity score $s(u,u')$. This fulfills the base case of our induction. Now, we move to our inductive step. Given a partial mapping $f^t : V_1^t \to V_2$ and associated alignment quality score AQ^t, we find the node pair (u,u') that gives the highest marginal increase in AQ. This is done as follows.

First, WAVE creates a vote matrix $\hat{s}(u,u')$, initialized to $\hat{s}(u,u') = \frac{s(u,u')}{|V_1|}$ for all $u \in V_1, u' \in V_2$. $\hat{s}(u,u')$ represents the marginal increase in AQ if we align node u to node u'. Second, WAVE selects the node pair (w,w') with the highest $\hat{s}(w,w')$, and aligns w to w'. Third, the aligned node pair (w,w') increments the weight of their neighbors with the marginal increase in AQ if that neighbor were to be aligned (this can be seen as a "voting" mechanism). That is, each node pair $(u,u') \in N_w \times N_{w'}$ receive a weighted vote $\frac{\beta(s(w,w')+s(u,u'))}{2\min(|E_1|,|E_2|)}$, normalized locally to $\frac{\beta(s(w,w')+s(u,u'))}{|V_1|}$, that updates its marginal increase in WEC, and 0 for the marginal increase in S_N. Finally, WAVE goes to the second step, repeating the process of selecting the node pair (w,w') with the highest $\hat{s}(w,w')$, until all nodes in V_1 are aligned. This alignment strategy greedily maximizes AQ. Supplementary Algorithm S1 describes our WAVE implementation that has $O(|V_1||V_2|\log(|V_1||V_2|) + |V_1|d_1d_2)$ time complexity, where d_1 and d_2 are the average degrees of G_1 and G_2, respectively.

Algorithm S1 WAVE algorithm.

procedure WAVE($G_1(V_1, E_1)$, $G_2(V_1, E_1)$, $s : |V_1| \times |V_2|$, β)
  Let $A = \varnothing$ be an empty set of aligned node pairs.
  Let $Q = \varnothing$ be an empty priority queue.
  Let $L_1 = \varnothing$ be an empty set of nodes.
  Let $L_2 = \varnothing$ be an empty set of nodes.
  Let $\hat{s} : |V_1| \times |V_2|$ be a matrix of zeros
  for $(u, u') \in V_1 \times V_2$ **do** ▷ Initialize priority queue
    $\hat{s}[u, u'] \leftarrow \frac{s[u,u']}{|V_1|}$
    $Q[(u, u')] \leftarrow \hat{s}[u, u']$
  while $Q \neq \varnothing$ **do** ▷ Main loop
    $(w, w') \leftarrow$ get_maximum_element(Q)
    $A \leftarrow A \cup \{(w, w')\}$
    $L_1 \leftarrow L_1 \cup \{w\}$
    $L_2 \leftarrow L_2 \cup \{w'\}$
    for $(u, u') \in N_w \times N_{w'}$ **do** ▷ Voting. N_w is the set of neighboring nodes of w, and similarly for $N_{w'}$
      $\hat{s}[v, v'] \leftarrow \frac{\beta(s(w,w')+s(u,u'))}{|V_1|}$
    for $(u, u') \in (N_w \setminus L_1) \times (N_{w'} \setminus L_2)$ **do** ▷ Update priority queue
      $Q[(u, u')] \leftarrow \hat{s}[u, u']$
    for $u \in V_1 \setminus L_1$ **do** ▷ Remove node pairs that cannot be aligned anymore
      remove_element($Q, (u, w')$)
    for $u' \in V_2 \setminus L_2$ **do**
      remove_element($Q, (w, u')$)

S1.2 DynaWAVE

DynaWAVE maximizes $\beta S_E + (1-\beta) S_N$, where S_E is the DWEC dynamic edge conservation measure, and $S_N = \frac{1}{|V_1|} \sum_{u \in V_1} s(u, u')$ is the node conservation measure with $s(u, u')$ being the node similarity between $u \in V_1$ and $u' \in V_2$, and β is a parameter between 0 and 1 that balances between the two conservation types. Node similarities $s(\cdot, \cdot)$ can be based on network topology only (e.g., dynamic graphlet-based node similarities), information external to network topology such as protein sequence similarities, or a combination of the two. Two node similarities, $s_1(u, v)$ and $s_2(u, v)$, between nodes u and v can be combined, for example, using the convex combination $\alpha s_1(u, v) + (1 - \alpha) s_2(u, v)$. Specifically in this paper, we use dynamic graphlet-based node similarities (Hulovatyy *et al.*, 2015) and set the β parameter to 0.5. Dynamic graphlets are an extension of static graphlets to the dynamic setting. While static graphlets capture the local topology of a node, dynamic graphlets capture how the local topology changes over time. We use the dynamic graphlet-based node similarity $s(\cdot, \cdot)$ for dynamic networks as described in Vijayan *et al.* (2017).

S1.2.1 DWEC

We continue our discussion of DWEC from Section 2 of the main paper.

DWEC is an extension of WEC from static NA to dynamic NA. Similar to DynaMAGNA++'s DS^3, DynaWAVE's DWEC also computes the conserved event time (CET) of the alignment (where the CET of the mapping of node pair (u, v) to node pair (u', v') is the amount of time during which both (u, v) and (u', v') are active, and the total alignment CET is the sum of CETs across all mapped node pairs (explained in detail below). However, unlike DS^3, DWEC favors a conserved event with similar end-nodes over an equally conserved event with dissimilar end-nodes. That is, when calculating DWEC, for each node pair $(u, v) \in V_1 \times V_1$ that is mapped to node pair $(u', v') \in V_2 \times V_2$, the CET of the node pair mapping is weighted by $s(u, u')$ and $s(v, v')$, the node similarity between nodes $(u, u') \in V_1 \times V_2$ and nodes $(v, v') \in V_1 \times V_2$, respectively.

To describe the above formally, first, we define a dynamic network and dynamic NA, then define CET, and then we define DWEC. Let a *dynamic network* $H(V, T)$ consist of a node set V and an event set T. An event is an interaction between nodes u and v from time t_s to t_e, and is represented as a 4-tuple (u, v, t_s, t_e). Let a *dynamic NA* be a one-to-one node mapping $f : V_1 \rightarrow V_2$ between dynamic networks $H_1(V_1, T_1)$ and $H_2(V_2, T_2)$, which produces the set of aligned node pairs $\{(v, f(v)) \mid v \in V_1\}$.

CET is an extension of the notion of a conserved edge from static NA to dynamic NA. Similar to how an edge $(u, v) \in E_1$ in static network $G_1(V_1, E_1)$ that is aligned to an edge $(u', v') \in E_2$ in static network $G_2(V_2, E_2)$ is conserved, we measure the CET of a the mapping of a node pair $(u, v) \in T_1$ in dynamic network $H_1(V_1, T_1)$ to a node pair $(u', v') \in T_2$ in dynamic network

$H_2(V_2, T_2)$. The CET of this node pair mapping measures the entire amount of time during which both node pairs are active at the same time. That is, the CET between (u, v) and (u', v') is

$$\mathrm{CET}((u,v),(u',v')) = \sum_{e \in T_{uv}} \sum_{e' \in T_{u'v'}} ct(e, e'),$$

where the conserved time $ct(e, e') = \max(0, \min(t_e, t'_e) - \max(t_s, t'_s))$ is the amount of time during which events $e = (u, v, t_s, t_e)$ and $e' = (u', v', t'_s, t'_e)$ are active at the same time, i.e., $ct(e, e')$ is the length of the overlap of the intervals $[t_s, t_e]$ and $[t'_s, t'_e]$. Vijayan *et al.* (2017) contains a more detailed explanation and justification for this extension of the notion of a conserved edge from static NA to dynamic NA.

Then, given two dynamic networks $H_1(V_1, T_1)$ and $H_2(V_2, T_2)$, assuming without loss of generality that $|V_1| \le |V_2|$, and a dynamic NA $f : V_1 \to V_2$ between them,

$$\mathrm{DWEC} = \sum_{(u,v) \in V_1 \times V_1} \frac{s(v, v')\,\mathrm{CET}((u,v),(u',v'))}{2\min(ta(H_1), ta(H_2))},$$

where $u' = f(u), v' = f(v)$, $\mathrm{CET}((u,v),(u',v'))$ is the CET of the mapping of node pair (u, v) to node pair (u', v') and is defined in (Vijayan *et al.*, 2017), and $ta(H)$ is the total amount of time during which events in dynamic network $H(V, T)$ are active, i.e., $ta(H) = \sum_{(u,v,t_s,t_e) \in T} (t_e - t_s) = \frac{1}{2} \sum_{(u,v) \in V \times V} \mathrm{CET}((u,v),(u,v))$.

S1.2.2 DynaWAVE alignment strategy

We continue our discussion of DynaWAVE's alignment strategy from Section 2 of the main paper. We extend the arguments for WAVE's alignment strategy from static NA to dynamic NA.

Above, we described DWEC when given two dynamic networks $H_1(V_1, T_1)$ and $H_2(V_2, T_2)$, and a dynamic NA $f : V_1 \to V_2$ between them. That is, f describes a mapping from V_1 to V_2 for all nodes $v \in V_1$. However, it is also possible to describe DWEC when given a partial mapping from V_1 to V_2, i.e., when given a partial mapping $f^* : V_1^* \to V_2$ where $V_1^* \subset V_2$. In order to motivate the DynaWAVE alignment strategy, we will now describe DWEC when given a partial mapping between two static networks. Given a partial mapping $f^t : V_1^t \to V_2$, where only the nodes in V_1^t are aligned, we are able to calculate the partial DWEC:

$$\mathrm{DWEC}^t = \sum_{(u,v) \in V_1^t \times V_1^t} \frac{s(v, v')\,\mathrm{CET}((u,v),(u',v'))}{2\min(ta(H_1), ta(H_2))}.$$

Now, what happens to DWEC^t when we align a new node $w \in V_1$ to the partial mapping? What is the marginal improvement in DWEC^t when we add a new node $w \in V_1$ that is mapped to node $w' \in V_2$? Formally, let $f^{t+1} : V_1^{t+1} \to V_2$ be the partial mapping when we add to V_1^t a new node $w \in V_1$ that maps to node $w' \in V_2$; that is, $V_1^{t+1} = \{w\} \cup V_1^t$ and $f^{t+1}(V^t) = \{w'\} \cup f^t(V^t)$ where $w' = f^{t+1}(w)$. Then, the marginal improvement of DWEC^t when adding the aligned node pair (w, w') to f^t is:

$$\begin{aligned}
\mathrm{DWEC}^{t+1} - \mathrm{DWEC}^t &= \sum_{(u,v) \in V_1^{t+1} \times V_1^{t+1}} \frac{s(v, v')\,\mathrm{CET}((u,v),(u',v'))}{2\min(ta(H_1), ta(H_2))} - \sum_{(u,v) \in V_1^t \times V_1^t} \frac{s(v, v')\,\mathrm{CET}((u,v),(u',v'))}{2\min(ta(H_1), ta(H_2))} \\
&= \sum_{u \in V_1^t} \frac{s(w, w')\,\mathrm{CET}((u,w),(u',w'))}{2\min(ta(H_1), ta(H_2))} + \sum_{v \in V_1^t} \frac{s(v, v')\,\mathrm{CET}((w,v),(w',v'))}{2\min(ta(H_1), ta(H_2))} \\
&= \sum_{u \in V_1^t} \frac{(s(w, w') + s(u, u'))\,\mathrm{CET}((u,w),(u',w'))}{2\min(ta(H_1), ta(H_2))} \qquad (\text{since } \mathrm{CET}((u,v),(u',v')) = \mathrm{CET}((v,u),(v',u'))) \\
&= \sum_{u \in V_1^t \wedge (u,u') \in N_w \times N_{w'}} \frac{(s(w, w') + s(u, u'))\,\mathrm{CET}((u,w),(u',w'))}{2\min(ta(H_1), ta(H_2))},
\end{aligned}$$

where N_u contains the neighbors of node u. So, the marginal improvement of DWEC^t weigh each conserved edge using the node similarities of w and its neighbors N_w, as well as the node similarities of w' and its neighbors $N_{w'}$, weighted by the conserved event time (CET) of the dynamic edges.

This suggests a greedy alignment strategy: given a partial alignment f^t, we may choose a new aligned pair (w, w') such that $\mathrm{DWEC}^{t+1} - \mathrm{DWEC}^t$ is maximized; furthermore, due to the above equation, we can see that this maximization procedure will depend only on the nodes that are already aligned and their neighbors. We then repeat this maximization process until all nodes in V_1 are aligned. We describe this alignment strategy below.

Note that as explained in Section 2 of the main paper, DynaWAVE maximizes $\beta S_E + (1-\beta)S_N$, where S_E is the DWEC dynamic edge conservation measure, and $S_N = \frac{1}{|V_1|}\sum_{u\in V_1} s(u,u')$ is the node conservation measure with $s(u,u')$ being the node similarity between $u \in V_1$ and $u' \in V_2$. Thus, the goal of DynaWAVE is to maximize the alignment quality measure $AQ = \beta DWEC + (1-\beta)S_N$. We now describe DynaWAVE's alignment strategy.

Given an initial empty mapping $f^0 : V_1^0 \to V_2$, where $V_1^0 = \varnothing$, the associated alignment quality score $\text{AQ}^0 = 0$. Then, the aligned node pair with the highest marginal increase in AQ is the node pair (u,u') with the highest node similarity score $s(u,u')$. This fulfills the base case of our induction. Now, we move to our inductive step. Given a partial mapping $f^t : V_1^t \to V_2$ and associated alignment quality score AQ^t, we find the node pair (u,u') that gives the highest marginal increase in AQ. This is done as follows.

First, DynaWAVE creates a vote matrix $\hat{s}(u,u')$, initialized to $\hat{s}(u,u') = \frac{s(u,u')}{|V_1|}$ for all $u \in V_1, u' \in V_2$. $\hat{s}(u,u')$ represents the marginal increase in AQ if we align node u to node u'. Second, DynaWAVE selects the node pair (w,w') with the highest $\hat{s}(w,w')$, and aligns w to w'. Third, the aligned node pair (w,w') increments the weight of their neighbors with the marginal increase in AQ if that neighbor were to be aligned (this can be seen as a "voting" mechanism). That is, each node pair $(u,u') \in N_w \times N_{w'}$ receive a weighted vote $\frac{\beta\,(s(w,w')+s(u,u'))\,\text{CET}((u,w),(u',w'))}{2\min(ta(H_1),ta(H_2))}$, normalized locally to $\frac{\beta\,(s(w,w')+s(u,u'))\,\text{CET}((u,w),(u',w'))}{|V_1|\max_{v,v'}(\text{CET}(T_{uv}\cup T_{u'v'}))}$, that updates its marginal increase in DWEC, and 0 for the marginal increase in S_N . Finally, DynaWAVE goes to the second step, repeating the process of selecting the node pair (w,w') with the highest $\hat{s}(w,w')$, until all nodes in V_1 are aligned. This alignment strategy greedily maximizes AQ. Supplementary Algorithm S2 describes our DynaWAVE implementation that has $O(|V_1||V_2|\log(|V_1||V_2|) + |V_1|e_1e_2)$ time complexity, where e_1 and e_2 are the average number of events in each node in H_1 and H_2, respectively.

Algorithm S2 DynaWAVE algorithm.

procedure DynaWAVE($G_1(V_1, T_1)$, $G_2(V_1, T_1)$, $s : |V_1| \times |V_2|$, β)
  Let $A = \varnothing$ be an empty set of aligned node pairs
  Let $Q = \varnothing$ be an empty priority queue
  Let $L_1 = \varnothing$ be an empty set of nodes
  Let $L_2 = \varnothing$ be an empty set of nodes
  Let $\hat{s} : |V_1| \times |V_2|$ be a matrix of zeros
  for $(u, u') \in V_1 \times V_2$ **do** ▷ Initialize priority queue
    $\hat{s}[u, u'] \leftarrow \frac{s[u,u']}{|V_1|}$
    $Q[(u, u')] \leftarrow \hat{s}[u, u']$
  while $Q \neq \varnothing$ **do** ▷ Main loop
    $(w, w') \leftarrow$ get_maximum_element(Q)
    $A \leftarrow A \cup \{(w, w')\}$
    $L_1 \leftarrow L_1 \cup \{w\}$
    $L_2 \leftarrow L_2 \cup \{w'\}$
    for $(u, u') \in N_w \times N_{w'}$ **do** ▷ Voting. N_w is the set of neighboring nodes of w, and similarly for $N_{w'}$
      $\hat{s}[u, u'] \leftarrow \frac{\beta\,(s(w,w')+s(u,u'))\,\mathrm{CET}((u,w),(u',w'))}{|V_1|\max_{v,v'}(\mathrm{CET}(T_{uv} \cup T_{u'v'}))}$ ▷ T_{uv} is the set of events between u and v, and similarly for $T_{u'v'}$. CET is defined in Supplementary Algorithm S3
    for $(u, u') \in (N_u \setminus L_1) \times (N_{u'} \setminus L_2)$ **do** ▷ Update priority queue
      $Q[(u, u')] \leftarrow \hat{s}[u, u']$
    for $u \in V_1 \setminus L_1$ **do** ▷ Remove node pairs that cannot be aligned anymore
      remove_element($Q, (u, w')$)
    for $u' \in V_2 \setminus L_2$ **do**
      remove_element($Q, (w, u')$)

Algorithm S3 Given a set of events, $T = \{(u, v, t_s, t_e)\}$, sorted from smallest to largest by start time t_s, this algorithm calculates the CET of the set. The CET is the total amount of time during which two events are active (i.e., where the time intervals overlap).

```
1:  procedure CET(T)
2:      if T = ∅ then
3:          return 0
4:      T_c ← 0
5:      (u, v, a, b) ← pop T
6:      while T ≠ ∅ do
7:          (u, v, c, d) ← pop T
8:          if b ≤ c then
9:              (a, b) ← (c, d)
10:         else if b > d then
11:             T_c ← T_c + (d − c)
12:             (a, b) ← (d, b)
13:         else
14:             T_c ← T_c + (b − c)
15:             (a, b) ← (b, d)
16:     return T_c
```

S2 Results and discussion

S2.1 Randomization

Here, we discuss the non-strict and strict randomization models that are used in Section 3 of the main paper, and was first used by Vijayan *et al.* (2017).

S2.1.1 Non-strict randomization model

We continue our discussion on the non-strict randomization model from Section 3 of the main paper. This randomization scheme proposed by Holme (2015) works as follows. Given the original dynamic network $H(V, T)$, to randomize to noise level p, first, we arbitrarily number all m events in the network as $T = \{e_1, e_2, \ldots, e_m\}$. Then, for each event e_i, with probability p we randomly select an event $e_{i'}$, and we rewire the two events. That is, given $e_i = (u, v, t_s, t_e)$ and $e_{i'} = (u', v', t'_s, t'_e)$, we either set $e_i = (u, v', t_s, t_e)$ and $e_{i'} = (u', v, t_s, t_e)$ with probability 0.5, or we set $e_i = (u, u', t_s, t_e)$ and $e_{i'} = (v, v', t_s, t_e)$ with probability 0.5. If the resulting e_i or $e_{i'}$ forms a loop or a multiple link, then we undo the rewiring and randomly select another event $e_{i'}$.

S2.1.2 Strict randomization model

We continue our discussion on the strict randomization model from Section 3 of the main paper. This randomization scheme proposed by Holme (2015) works as follows. Given the original dynamic network $H(V, T)$, to randomize to noise level p, first, we arbitrarily number all m events in the network as $T = \{e_1, e_2, \ldots, e_m\}$. Then, for each event e_i, with probability p we randomly select another event $e_{i'}$, $i' \neq i$, and swap the time stamps of the two events.

S2.2 Synthetic networks

We continue our discussion of synthetic networks from Section 3 of the main paper. We create synthetic networks using three dynamic network models that were described in Hulovatyy *et al.* (2015): geometric gene duplication model with probability cutoff (GEO-GD, parameter $p = 0.3$, linear node arrival, Pržulj *et al.* (2010)), scale-free gene duplication model (SF-GD, parameters $p = 0.3, q = 0.7$, exponential node arrival, Vazquez *et al.* (2002)), and a social network evolution model (SNE, parameters $\lambda = 0.032, \alpha = 0.8, \beta = 0.002$, quadratic node arrival, Leskovec *et al.* (2008)). Table S2 shows the details of the generated synthetic networks.

Supplementary Tables

Algorithms	Parameters
DynaMAGNA++	m=DS3, p=15000, n=10000, a=0.5, $s(\cdot,\cdot)$ = dynamic GDV node similarity
MAGNA++	m=S3, p=15000, n=10000, a=0.5, $s(\cdot,\cdot)$ = static GDV node similarity
DynaWAVE	beta=0.5, $s(\cdot,\cdot)$ = dynamic GDV node similarity
WAVE	beta=0.5, $s(\cdot,\cdot)$ = static GDV node similarity
DynaMAGNA++ (DWEC)	m=DWEC, p=15000, n=10000, a=0.5, $s(\cdot,\cdot)$ = dynamic GDV node similarity
MAGNA++ (WEC)	m=WEC, p=15000, n=10000, a=0.5, $s(\cdot,\cdot)$ = static GDV node similarity

Table S1: Method parameters that we use in our study. We use parameters recommended in the methods' original publications. We use similar corresponding parameters as the other studies for DynaWAVE. $s(\cdot,\cdot)$ indicates the node similarity values given to the method. Details on both the dynamic and static GDV node similarity measures are described in Vijayan *et al.* (2017).

Network model	Num. nodes	Num. events
GEO-GD	100	296.00 (136.57)
GEO-GD	200	480.00 (237.46)
GEO-GD	400	808.00 (158.76)
GEO-GD	800	2094.67 (106.59)
GEO-GD	1600	5247.33 (1096.29)
GEO-GD	2400	9506.00 (1296.39)
GEO-GD	3200	9843.33 (1038.07)
GEO-GD	4000	13646.67 (883.46)
SF-GD	100	338.67 (78.85)
SF-GD	200	799.33 (105.10)
SF-GD	400	1440.67 (508.39)
SF-GD	800	2539.33 (216.97)
SF-GD	1600	4838.67 (570.02)
SF-GD	2400	7112.00 (42.33)
SF-GD	3200	10112.67 (1352.24)
SF-GD	4000	11067.33 (915.94)
SNE	100	275.33 (16.17)
SNE	200	592.67 (5.03)
SNE	400	1224.67 (32.02)
SNE	800	2513.33 (98.07)
SNE	1600	5184.00 (24.58)
SNE	2400	7879.33 (20.03)
SNE	3200	10510.00 (81.90)
SNE	4000	13095.33 (185.30)

Table S2: Details of synthetic networks generated using the GEO-GD, SF-GD, and SNE network models for varying network sizes. The average number of events (with standard deviation in brackets) is shown for each network model and each network size.

Supplementary Figures

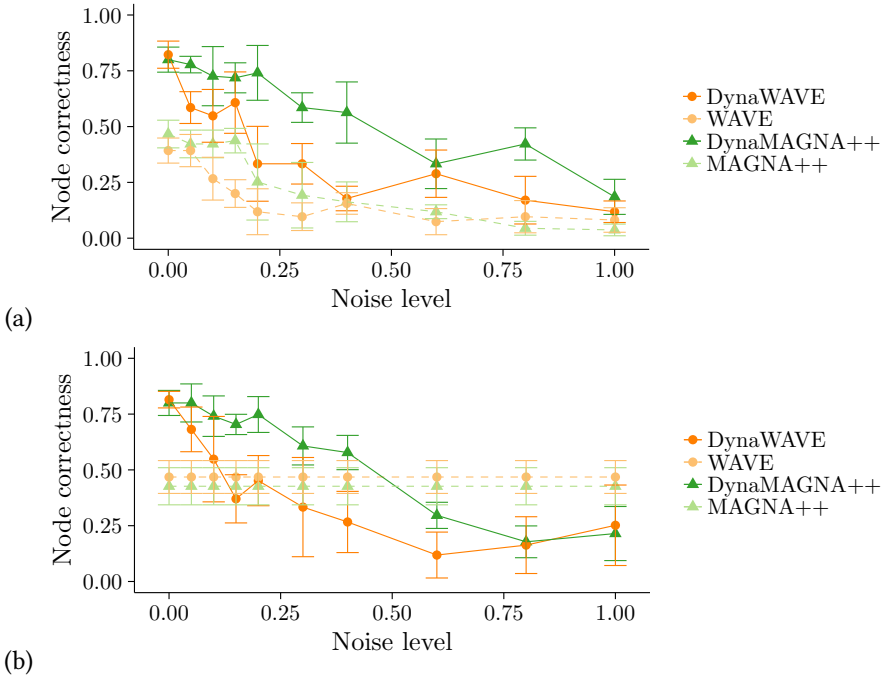


Figure S1: Node correctness (NC) of DynaWAVE, WAVE, DynaMAGNA++, and MAGNA++ as a function of noise level while aligning the original Zebra proximity network to randomized (noisy) versions of the original network. Two randomization models are used: (a) the non-strict randomization model that randomly rewires events in the network up to the given percentage (noise level) and does not conserve the structure of the flattened version of the original dynamic network, and (b) the strict randomization model that conserves all structure of the flattened version of the original dynamic network and only randomly "shuffles" the given percentage (noise level) of its event time stamps.

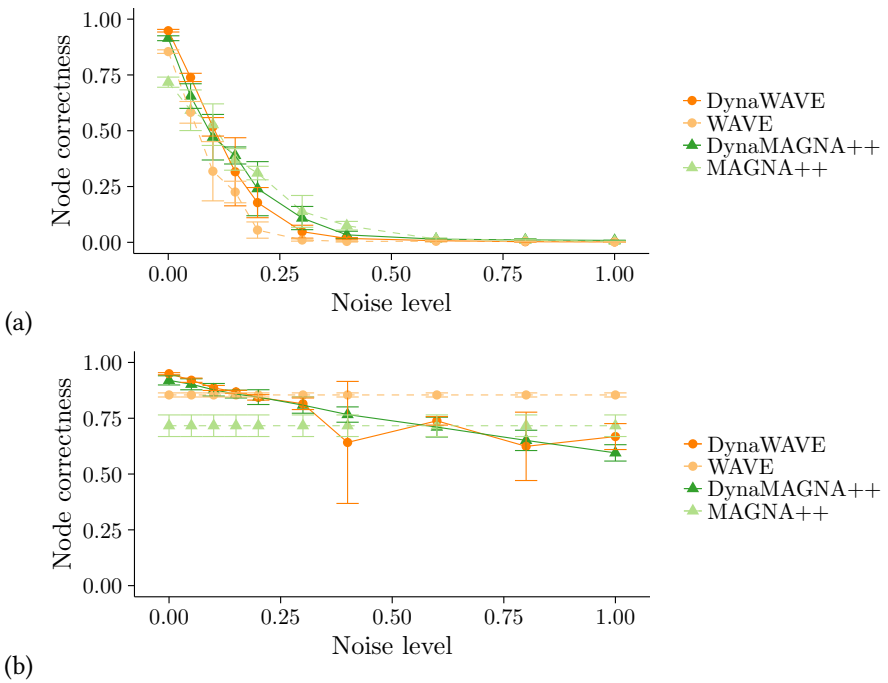
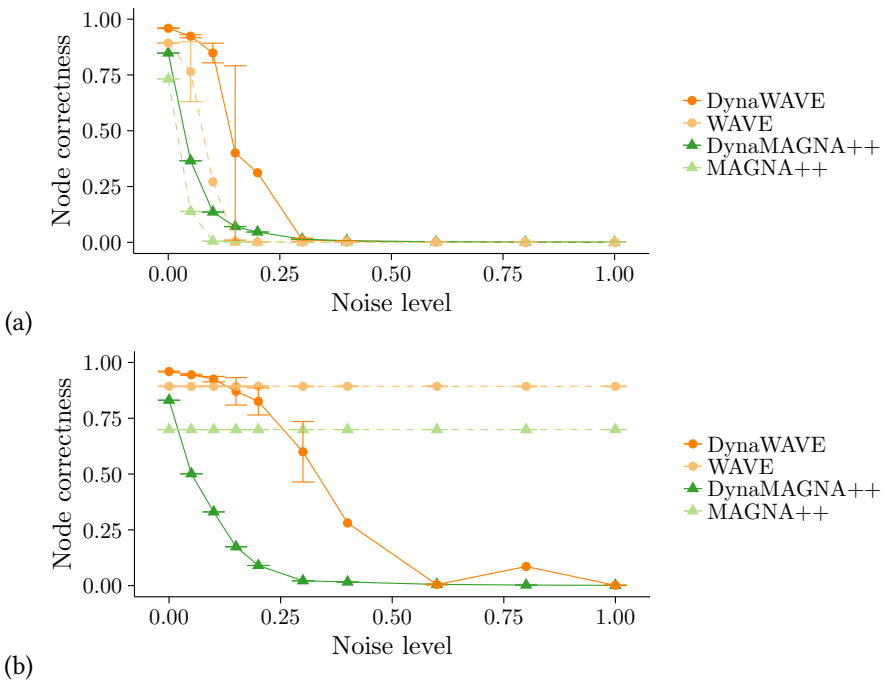


Figure S2: Node correctness (NC) of DynaWAVE, WAVE, DynaMAGNA++, and MAGNA++ as a function of noise level while aligning the original Yeast protein interaction network to randomized (noisy) versions of the original network. Two randomization models are used: (a) the non-strict randomization model that randomly rewires events in the network up to the given percentage (noise level) and does not conserve the structure of the flattened version of the original dynamic network, and (b) the strict randomization model that conserves all structure of the flattened version of the original dynamic network and only randomly "shuffles" the given percentage (noise level) of its event time stamps.



(a)

(b)

Figure S3: Node correctness (NC) of DynaWAVE, WAVE, DynaMAGNA++, and MAGNA++ as a function of noise level while aligning the original Human aging protein interaction network to randomized (noisy) versions of the original network. Two randomization models are used: (a) the non-strict randomization model that randomly rewires events in the network up to the given percentage (noise level) and does not conserve the structure of the flattened version of the original dynamic network, and (b) the strict randomization model that conserves all structure of the flattened version of the original dynamic network and only randomly "shuffles" the given percentage (noise level) of its event time stamps.

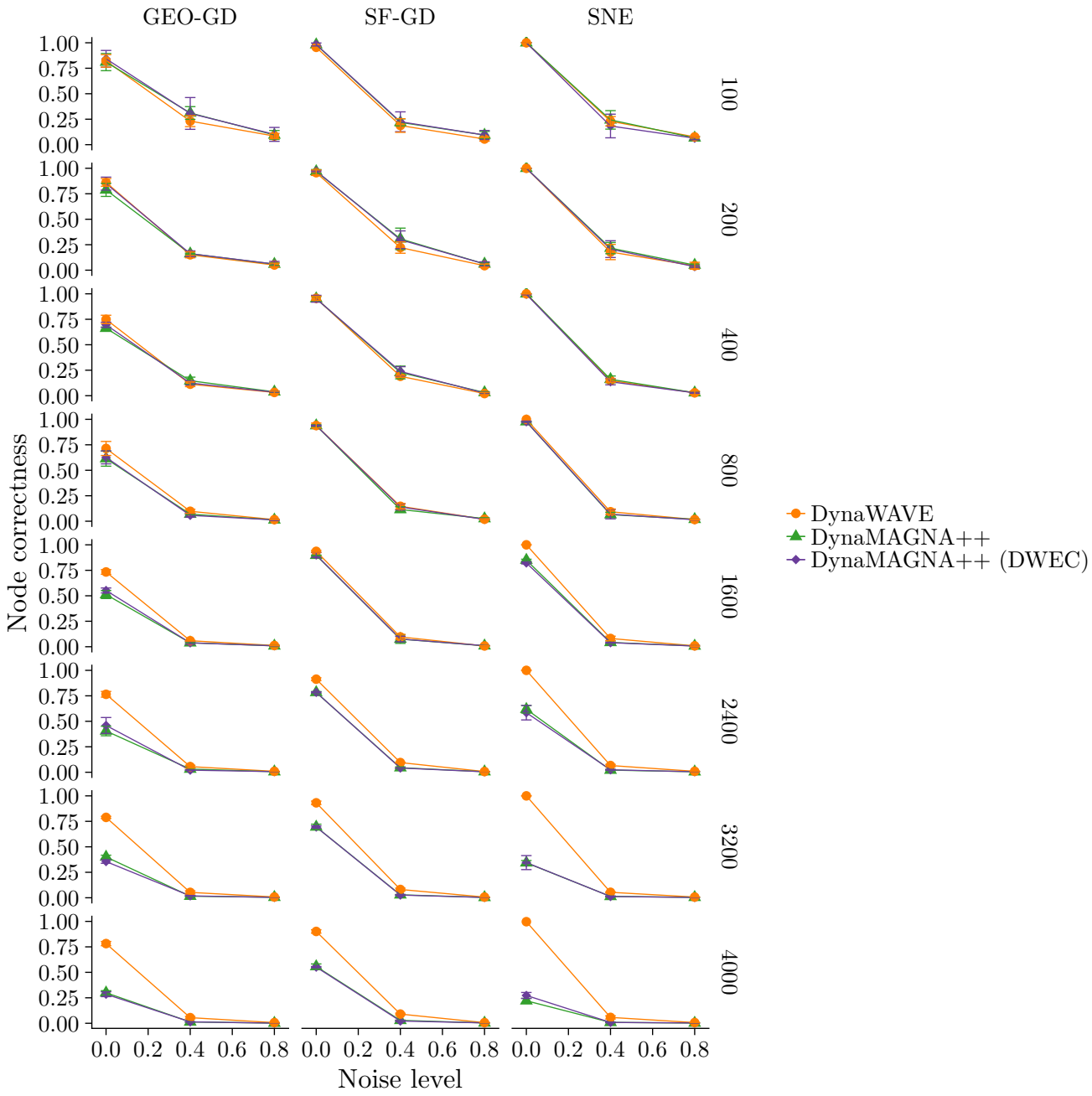


Figure S4: Node correctness (NC) of DynaWAVE, DynaMAGNA++, and DynaMAGNA++ (DWEC) as a function of noise level while aligning an original synthetic network to randomized (noisy) versions of the original network. NC is averaged for each network model (GEO-GD, SF-GD, SNE), for each network size (100, 200, 400, 800, 1600, 2400, 3200, 4000) at a given noise level. The non-strict randomization model that rewires events in the network and does not conserve structure of the flattened version of the original dynamic network is used.

Funding

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-16-1-0147.

References

Holme, P. (2015). Modern temporal network theory: a colloquium. *The European Physical Journal B*, **88**(9), 1–30.

Hulovatyy, Y., Chen, H., and Milenković, T. (2015). Exploring the structure and function of temporal networks with dynamic graphlets. *Bioinformatics*, **31**(12), 171–180.

Leskovec, J., Backstrom, L., Kumar, R., and Tomkins, A. (2008). Microscopic evolution of social networks. In *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 462–470.

Milenković, T. and Pržulj, N. (2008). Uncovering biological network function via graphlet degree signatures. *Cancer Informatics*, **6**, 257–273.

Pržulj, N., Kuchaiev, O., Stevanović, A., and Hayes, W. (2010). Geometric evolutionary dynamics of protein interaction networks. In *Proc. of the Pacific Symposium Biocomputing*, pages 4–8.

Vazquez, A., Flammini, A., Maritan, A., and Vespignani, A. (2002). Modeling of protein interaction networks. *ComPlexUs*, **1**, 38–44.

Vijayan, V., Critchlow, D., and Milenković, T. (2017). Alignment of dynamic networks. *Bioinformatics*, **33**(14), i180–i189.