Journal of Complex Networks (2018) **00**, 1–23
doi: 10.1093/comnet/cny030

Optimizing state change detection in functional temporal networks through dynamic community detection

MICHAEL VAIANA
Department of Mathematics and CDSE Program
University at Buffalo, SUNY, Buffalo, NY 14260, USA

ETHAN M. GOLDBERG
Division of Neurology, Department of Pediatrics
Children's Hospital of Philadelphia, Philadelphia, PA 19104, USA and
Departments of Neurology and Neuroscience
Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA 19104, USA

AND

SARAH F. MULDOON[†]
Department of Mathematics, CDSE Program, and Neuroscience Program
University at Buffalo, SUNY, Buffalo, NY 14260, USA
[†]Corresponding author. Email: smuldoon@buffalo.edu

Edited by: Mason Porter

[Received on 18 May 2018; editorial decision on 30 October 2018; accepted on 31 October 2018]

Dynamic community detection provides a coherent description of network clusters over time, allowing one to track the growth and death of communities as the network evolves. However, modularity maximization, a popular method for performing multilayer community detection, requires the specification of an appropriate null network as well as resolution and interlayer coupling parameters. Importantly, the ability of the algorithm to accurately detect community evolution is dependent on the choice of these parameters. In functional temporal networks, where evolving communities reflect changing functional relationships between network nodes, it is especially important that the detected communities reflect any state changes of the system. Here, we present analytical work suggesting that a uniform null network provides improved sensitivity to the detection of small evolving communities in temporal networks with positive edge weights bounded above by 1, such as certain types of correlation networks. We then propose a method for increasing the sensitivity of modularity maximization to state changes in nodal dynamics by modelling self-identity links between layers based on the self-similarity of the network nodes between layers. This method is more appropriate for functional temporal networks from both a modelling and mathematical perspective, as it incorporates the dynamic nature of network nodes. We motivate our method based on applications in neuroscience where network nodes represent neurons and functional edges represent similarity of firing patterns in time. We show that in simulated data sets of neuronal spike trains, updating interlayer links based on the firing properties of the neurons provides superior community detection of evolving network structure when groups of neurons change their firing properties over time. Finally, we apply our method to experimental calcium imaging data that monitors the spiking activity of hundreds of neurons to track the evolution of neuronal communities during a state change from the awake to anaesthetized state.

Keywords: dynamic networks; state change; modularity; community detection; functional networks.

1. Introduction

Networks are excellent data structures for capturing the pairwise interactions between a collection of objects. However, when these interactions are dynamic, traditional network approaches require some form of data reduction to reduce the full range of interactions to a single summary. As a result, the final network may or may not be a good representative of the underlying system. A more suitable data structure for dynamic interactions is a *temporal network* [1, 2]. Temporal networks can be represented in multiple ways, but here we focus on the snapshot representation of temporal networks [3], which is a time ordered collection of networks organized into layers together with *interlayer edges* connecting nodes between layers. Temporal networks are increasingly being used to model time dependent properties of complex systems since no data loss is incurred as in the case of static networks [4–8].

A particularly important class of temporal networks are functional networks [9], in which nodes represent a dynamic unit and edges are given by a measure of functional similarity (see Fig. 1). For example, in brain networks, nodes may represent neurons and edges are measured via synchronization between temporal spiking activity patterns [10]. Since the nodes in a functional temporal network are themselves dynamic, the system may exhibit state changes where nodes switch between functional states. Such a state change is depicted in Fig. 1 and occurs between time layers 2 and 3. These state changes effect the properties and structure of the network, and it is important to develop network measures that are sensitive to such changes [11]. Intuitively, one would expect interlayer edges to play a major role in temporal networks, as they model the strength of connections between nodes through time. However, it is typically assumed that interlayer edges exist only between a node and itself in the next layer, representing a self-identity link through time, and more importantly, the standard assumption is that interlayer edge weights are all set to the same constant value through time [4, 12–14]. While the latter assumption has made temporal networks easier to deploy in practice (since tuning a single parameter for interlayer edge weights is significantly easier than tuning all possible edge weights individually), it is insufficient at properly modelling functional networks that exhibit state changes where the properties of a node change over time and the similarity of a node to itself in the previous layer is not always constant.

One important property of a functional temporal network that is especially sensitive to state changes is the community structure [15–17]. In temporal networks, communities can and do evolve through time, and nodal state changes can have drastic effects on this evolving community structure. A popular method of community detection is modularity maximization [18, 19], and this method has been extended to temporal networks [12]. While multilayer modularity maximization has been shown to provide a relatively efficient and effective description of dynamic community structure [4, 20–23] the multilayer modularity function has three parameters that must be specified: the resolution parameter, the interlayer coupling parameter and the null network. While some work has investigated how to choose these parameters in certain settings [14, 22, 24, 25], how to optimally select these parameters remains an active area of research. In this article, we present work guiding the selection of these important parameters in order to provide improved sensitivity to the detection of state changes in dynamic community structure.

First, expanding upon recent work that identifies an upper bound on the interlayer coupling parameter that limits the ability of multilayer modularity to detect certain changes in community structure across layers [26], we show that for functional temporal networks whose edge weights are bounded above by 1, using a uniform null network (as opposed to the commonly used Newman–Girvan null network) is preferable for detecting evolving communities. We then propose a method to set interlayer edge weights in functional temporal networks based on a measure of temporal self-similarity. In accordance with intuition, we lower the interlayer edge weight between a node and itself at the next time point when the functional similarity between that node and itself is low. This gives a principled method of setting interlayer edge weights that further increases the sensitivity of modularity to detecting community changes. Finally, we

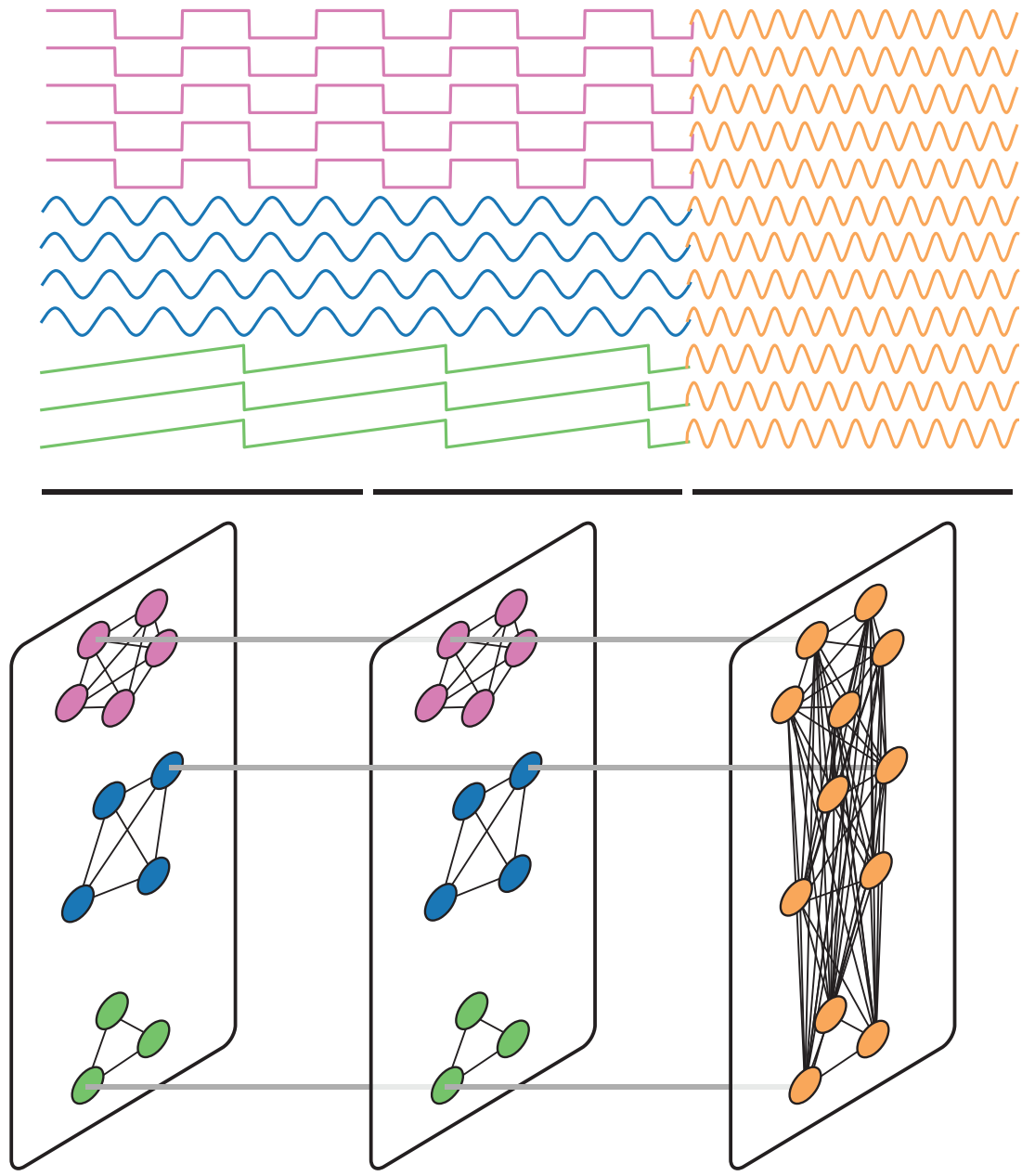

FIG. 1. Functional temporal network. The mapping of the activity of 12 functioning units to a temporal network. Colour indicates community assignment. At the top there are 12 functioning nodes which initially are partitioned into 3 groups each with a unique function. The system then undergoes a state change and the activity of the nodes synchronizes. Black bars indicate windows in which a measure of similarity between function is measured. The bottom is the temporal network with nodes representing the functioning units and edges representing the similarity in function in the given time window. For visual clarity only 9 interlayer edges are drawn connecting 3 nodes over time. Intuitively, the interlayer edge weight between layers 2 and 3 should be low to reflect the state change.

show how performing an essentially parameter free consensus, in combination with the methodology proposed above, gives nearly as accurate community detection results than those found with optimal choice of parameters when applied to simulated data representing neural activity with an implanted evolving community structure containing multiple state changes.

The article is organized as follows. In Section 2, we briefly review temporal networks and the temporal modularity function. In Section 3, we discuss parameter selection for the modularity function and present our method of setting interlayer edge weights. In Section 4, we test our method on simulated data and show that it gives improved sensitivity to the modularity function. In Section 5, we apply our method to real-world neuronal spike train data to monitor the evolution of neuronal communities during the transition to and from the awake and anaesthetized states.

2. Communities in temporal networks

2.1 *Temporal networks and notation*

A temporal network is a network $\mathcal{M} = (V, w, l)$ with V the vertices, $w : V \times V \to R$ a weight function on the edges, and $l : V \to N \times L$ a labelling function where N and L denote the nodes and layers of the network, respectively. The function l identifies a vertex v with a node-layer pair, that is $l(v) = (i, t)$ indicates that v is the node i at time t, and we denote this as vertex i_t. The edges are implicitly defined via w so that $w(i_r, j_t)$ is the weight of the edge between i_r and j_t. The edge between i_r and j_t is called an *intralayer edge* if $r = t$ and an *interlayer edge* if $r \neq t$. We assume the interlayer coupling is diagonal and ordinal meaning there is no interlayer edge between i_r, j_t unless $i = j$ and $r = t \pm 1$. We let A_{ijt} be the intralayer edge between node i_t and j_t so A_{ijt} represents the adjacency matrix at time t. Similarly, we let B_{it} be the interlayer edge between node i_t and i_{t+1}.

2.2 *Modularity in temporal networks*

Let $\mathcal{T}$ be a temporal network with L many time layers. A community, C, in a temporal network is any subset of the vertices of the network, and importantly, communities can span multiple layers through time. Let $P = \{C_1, C_2, \ldots, C_k\}$ be a partition of the vertices of the network where each C_i is a community. The modularity function measures how well a given partition, P, captures any underlying community structure of the network. It takes three parameters: an intralayer resolution parameter, γ, an interlayer coupling parameter, B_{it}, a null network, R, and is given by

$$Q(P) = \sum_{t}^{L} \sum_{ij} (A_{ijt} - \gamma R_{ijt}) \delta(c_{i_t}, c_{j_t}) + \sum_{t}^{L-1} \sum_{i} 2B_{it} \delta(c_{i_t}, c_{i_{t+1}}), \tag{1}$$

where c_{i_t} is the community assignment of i_t in P and δ is the Kroneker delta function [12]. For now, we will make the common assumption that $B_{it} = \omega$, where ω is a constant value across all nodes and layers, but we will relax this assumption later. The partition that maximizes the modularity function, Q, consists of the putative communities.

3. Optimal modularity parameters in temporal networks

3.1 *Multilayer resolution limit and null networks*

The temporal modularity function is highly sensitive to its three parameters, the resolution parameter, γ, the interlayer coupling parameter, ω, and the null network, R. It is important to understand which choice

of parameters is suitable for different types of networks. In [26], it was shown that all three parameters are related through a multilayer resolution limit: an upper bound on the interlayer coupling parameter which determines when modularity can detect mergers of communities. Here, we examine how the multilayer resolution limit is affected by the choice of null network by comparing this limit for two popular choices of null networks: the uniform null network (U) and the classic Newman–Girvan null network (NG). We will show that, due to the resolution limit, in networks with positive edge weights bounded above by 1, it is preferable to use the U null network as opposed to the traditional NG null network.

Let $\mathcal{T}$ be a temporal network with L many time layers, N many nodes on each layer, and positive edge weights. Let K be a subset of the N nodes and assume that at time t the nodes of K form communities $C_1, C_2, \ldots, C_r$ and that at time $t+1$ these all merge into a single community. For example, the three communities in Fig. 1 merge together in layer 3 of the temporal network.

It was shown in [26] that the merger of the communities cannot be detected by modularity if $\omega > \Omega_t$, where

$$\Omega_t = \sum_{\substack{ij \in K \\ \delta(c_{i_t}, c_{j_t})=0}} \frac{1}{2\theta}(\gamma R_{ijt} - A_{ijt}) \tag{2}$$

and $\theta = |K| - |C_{\max}|$ where $C_{\max}$ is the largest of the communities. Therefore, Ω_t acts as a type of multilayer resolution limit on detecting changes in community structure across layers. We refer to Ω_t throughout as the upper bound on interlayer coupling or simply the upper bound. Notice that all quantities in Equation 2 are with respect to layer t. For notational convenience, we drop the subscript t when it is understood we are working with a fixed layer.

Specializing to a null network lets one compute Ω_t more explicitly. Assume we are working with a fixed layer t, and in this layer there are n many communities $C_1, \ldots, C_n$ that merge together at time $t+1$. Define $\kappa_i = \sum_j A_{ij}$ to be the degree of node i and $2m = \sum_i \kappa_i$, where m is the number of edges in the network. The NG null network is then given by $R_{ij} = \frac{\kappa_i \kappa_j}{2m}$. The U null network is given by $R_{ij} = \langle A \rangle$ for all i, j where $\langle A \rangle$ is the mean value of A. Explicitly, we have $\langle A \rangle = \frac{2m}{|N|^2}$.

In [26], it was shown that for the NG null network, the upper bound is given by

$$\Omega_{\mathrm{NG}} = \frac{1}{2\theta}\left(\frac{\gamma}{2m}\sum_{i \neq j} d_i d_j - \sum_i e_i\right), \tag{3}$$

where $d_i = \sum_{j \in C_i} \kappa_j$ is the degree of community C_i, and e_i is the number of edges that link a node within C_i to a node outside of C_i. Recomputing Ω with respect to the U null network gives

$$\Omega_{\mathrm{U}} = \frac{1}{2\theta}\left(\gamma \sum_{i \neq j} |C_i||C_j| - \sum_i e_i\right), \tag{4}$$

where $|C_i|$ is the number of nodes in community C_i. Interestingly, for the U null network, the term that contributes positively to Ω does not depend on the edges within the communities and instead only depends on number of nodes within the communities.

We now give conditions on the community structure that guarantee $\Omega_U > \Omega_{\mathrm{NG}}$. When this inequality holds, the upper bound for the U null network is larger than that of the NG network, thus implying there

is a wider range of ω for which the communities $C_1, \ldots, C_n$ can be detected. We first prove a simple lemma that lets us approximate Ω_{NG} with $\hat{\Omega}_{\text{NG}}$ where $\Omega_{\text{NG}} \leq \hat{\Omega}_{\text{NG}}$.

Lemma 3.1.1. Let $\mathcal{T}$ be a temporal network such that communities $C_1, \ldots, C_n$ on layer t merge together on layer $t+1$. Let K be the nodes of the communities $C_1 \ldots C_n$. Then

$$\Omega_{\text{NG}} \leq \frac{1}{2\theta}\left(\gamma 2k - \sum_i e_i\right) \equiv \hat{\Omega}_{\text{NG}}$$

where $2k = \sum_{ij \in K} A_{ijt}$.

Proof. Since K is a subset of nodes of the network we have $2k \leq 2m$, and thus

$$\Omega_{\text{NG}} = \frac{1}{2\theta}\left(\frac{\gamma}{2m}\sum_{i \neq j} d_i d_j - \sum_i e_i\right) \tag{5}$$

$$\leq \frac{1}{2\theta}\left(\frac{\gamma}{2m}(2k)^2 - \sum_i e_i\right) \tag{6}$$

$$\leq \frac{1}{2\theta}\left(\gamma 2k - \sum_i e_i\right) \tag{7}$$

$$\equiv \hat{\Omega}_{\text{NG}}. \tag{8}$$

□

Theorem 3.1.2. Let $\mathcal{T}$ be a temporal network such that communities $C_1, \ldots, C_n$ on layer t merge together on layer $t+1$. Let K be the nodes of the communities $C_1 \ldots C_n$, and let $\langle K \rangle$ be the average edge weight of those nodes. Then, $\Omega_U > \Omega_{\text{NG}}$ if

$$\langle K \rangle < 1 - \frac{\sum_i |C_i|^2}{|K|^2}.$$

Proof. Our aim is to show that $\Omega_U > \Omega_{\text{NG}}$. By Lemma 3.1.1, it will suffice to show that $\Omega_U > \hat{\Omega}_{\text{NG}} > \Omega_{\text{NG}}$. We compute

$$\Omega_U - \hat{\Omega}_{\text{NG}} = \frac{1}{2\theta}\left(\gamma \sum_{i \neq j} |C_i||C_j| - \sum_i e_i\right) - \frac{1}{2\theta}\left(\gamma 2k - \sum_i e_i\right) \tag{9}$$

$$= \frac{\gamma}{2\theta}\left(\sum_{i \neq j} |C_i||C_j| - 2k\right) \tag{10}$$

$$= \frac{\gamma}{2\theta}\left(|K|^2 - \sum_i |C_i| - |K|^2\langle K \rangle\right). \tag{11}$$

To obtain the last equality we used the fact that $2k = |K|^2\langle K\rangle$ and that

$$|K|^2 = \sum_{i,j} |C_i||C_j| = \sum_{i\neq j} |C_i||C_j| + \sum_i |C_i|^2.$$

From the computation, we see that $\Omega_U > \hat{\Omega}_{\mathrm{NG}}$ if and only if

$$|K|^2 - \sum_i |C_i| - |K|^2\langle K\rangle > 0$$

and solving for $\langle K\rangle$ completes the proof. □

The content of the theorem says that if the average edge weight of the nodes that merge together is bounded by $1 - \frac{\sum_i |C_i|^2}{|K|^2}$, then we can conclude that $\Omega_U > \Omega_{\mathrm{NG}}$. In any network whose edge weights are bounded by 1, the average edge value will also be bounded by 1. Since $|K|^2 = \sum_{i\neq j} |C_i||C_j| + \sum_i |C_i|^2$, the quantity $\frac{\sum_i |C_i|^2}{|K|^2} \leq 1$. How much less than one will depend upon the topologies of the communities that merge.

Example 3.1.3. Assume the communities $C_1, \ldots, C_n$ from Theorem 3.1.2 all have the same size, s. Then $\Omega_{\mathrm{U}} > \Omega_{\mathrm{NG}}$ if $\langle K\rangle < 1 - \frac{1}{n}$. We can see this by noting that $\sum_i |C_i|^2 = ns^2$ and $|K|^2 = (ns)^2$, and thus by the theorem,

$$\langle K\rangle < 1 - \frac{ns^2}{(ns)^2} = 1 - \frac{1}{n}.$$

Therefore, in this example, if the average edge weight of the nodes that merge together is less than $1 - \frac{1}{n}$, it is preferable to use a uniform null network model.

This result has important practical implications for functional temporal networks, as many measures of similarity between time series are bounded between 0 and 1. This finding is also is in agreement with Bazzi *et al.* [14], who argued that in multilayer correlation networks, the U null network may be more suitable than the classic NG null network. (However, one should also see [27] and [28] for more detailed discussions of how to properly determine suitable null models in correlation networks.)

There are a few final points that need to be made clear. First, the converse of Theorem 3.1.2 does not hold, that is if $\langle K\rangle > 1 - \frac{\sum_i |C_i|^2}{|K|^2}$ then nothing can be said about which bound, Ω_{U} or Ω_{NG}, is larger. Second, the average edge weight, $\langle K\rangle$, is the average weight of the edges taken over all nodes in K. Since we are assuming there are multiple communities in K, the connections between communities will necessarily be weak. Thus, even in networks whose edge weights are bounded above by one, we expect the average value of the edge weights to be much lower for most community structures. It should also be noted that ensuring $\Omega_U > \Omega_{\mathrm{NG}}$ does not itself mitigate the problem of the multilayer resolution limit. However, it does provide a wider range of possible values of ω for which $\omega < \Omega$.

3.2 *Dynamic interlayer edges*

Community detection in functional temporal networks presents several challenges. We would like to model the underlying system in a way that allows for improved sensitivity to the detection of changes

in state or behaviour. The first step in improving such sensitivity is to leverage the interlayer edges by letting them reflect a measure of self-similarity in nodes instead of being held constant (i.e. we will no longer assume $B_{it} = \omega$). Such a measure of self-similarity is often available in functional networks, and we assume that the self-similarity of node, i_t, to itself one time step later, i_{t+1}, is given by $s(i_t, i_{t+1})$. Practically, one would define this measure based on some sort of nodal feature that quantifies nodal dynamics and that that differs in different states of the system. In the following sections, we will give examples analyzing neuronal spike train data where self-similarity is assessed by the change of firing rate of neurons in sequential layers. However, we emphasize that the following procedure can be applied to any general measure of self-similarity and how to define this measure will necessarily be application dependent.

Our method is to first choose ω_{global} and set $B_{it} = \omega_{\text{global}}$ for all i and t (recall B_{it} gives the interlayer edge between i_t and i_{t+1}). We then update B_{it} on a node-by-node, layer-by-layer basis, depending on the value of $s(i_t, i_{t+1})$. If we let $\rho \leq 1$ be a percentage, then our method is summarized as

1. Choose ω_{global} and set $B_{it} = \omega_{\text{global}}$ for all i and t.
2. For each i and t if $s(i_t, i_{t+1})$ is 'small', then set $B_{it} = \rho\omega_{\text{global}}$.

We refer to this process as *updating interlayer edges*.

An essential step in this process is the determination of when the self-similarity, $s(i_t, i_{t+1})$, is considered to be 'small'. While this definition is application dependent, we propose two methods for defining thresholds that quantify whether or not to update an edge. The first method, which we refer to as a *local threshold*, assumes that in general, one expects similar types of changes to occur over time (between layers). In this case, one can assess the value of a nodal feature and calculate how much this feature changes between two sequential layers. One then chooses to update the interlayer edges for nodes whose change lies outside of X standard deviations from the mean of the distribution. By varying the value of X, one can choose to update fewer or greater numbers of edges. This process can be computed locally at each pair of sequential layers. The second method, which we refer to as a *global threshold*, calculates the change in nodal features between all nodes across all sequential layers, and then chooses to update an interlayer edge if the change in nodal feature lies within the top $Y\%$ of the global distribution of changes. This method is useful in cases where one does not expect similar types of change to exist between layers over time. One could also determine other criteria for choosing how to perform updates such as updating each edge proportionally with respect to the self-similarity of the node in adjacent layers, and we encourage future work to investigate such options.

3.3 *Parameter consensus*

Maximizing the modularity function is computationally infeasible so in practice one uses a heuristic, for example the Louvain algorithm [29, 30]. It has been shown that the modularity landscape has a large number of locally optimal partitions [31] which can correspond to significantly different community assignments. To combat this dilemma, one can run a heuristic many times and form a consensus partition over the different runs [24, 32]. The idea is that the true core structure should be detected by the algorithm over most of the runs while weaker structure may only get detected a few times. Using a consensus can then determine the structure that is consistently found.

There are several different approaches to applying a consensus algorithm [22, 24, 32–35], but here we focus on methods that allow us to incorporate our method of updating interlayer edges. Because the algorithm for maximizing modularity is stochastic, one may choose a fixed pair of parameters (γ_0, ω_0)

and run the algorithm many times with this fixed choice. Alternatively, one can chose a set of pairs $\{(\gamma_0, \omega_0), (\gamma_1, \omega_1), \ldots (\gamma_n, \omega_n)\}$ and run the algorithm once for each of these pairs. In this formulation, the resulting structure one obtains will be consistently found over different spatial resolutions (due to varying γ) and temporal resolutions (due to varying ω).

To understand the effects of our method of updating interlayer edges, we run community detection in four different ways:

1. Fixed consensus: We choose a fixed (γ, ω) and run a consensus on multiple runs of the community detection algorithm. Interlayer edges are not updated.
2. Sweep consensus: We choose a grid of points in the γ, ω plane and run a consensus over a single run of the community detection algorithm for each combination of these parameters. Interlayer edges are not updated.
3. Fixed with updates: We choose a fixed $(\gamma, \omega_{\text{global}})$ and run a consensus on multiple runs of the community detection algorithm. However, during each run, we additionally apply our update procedure from the previous section.
4. Sweep with updates. We choose a grid of points in the $\gamma, \omega_{\text{global}}$ plane run a consensus over a single run of the community detection algorithm for each combination of these parameters. However, for each choice of $(\gamma, \omega_{\text{global}})$, we additionally apply our update procedure from the previous section.

4. Optimal parameter selection in dynamic simulated neuronal data

We now apply our proposed methodology to simulated neural spike train data with multiple embedded state changes to show that the use of a uniform null network model and dynamically updating interlayer edges gives superior performance to traditional methods for detecting evolving community structure. We stochastically generate 100 synthetic temporal networks with the same planted community structure and attempt to recover these communities via multilayer modularity maximization. Since our networks are created synthetically, we have access to the true community structure. To compare the results of a community consensus with the true community structure, we measure the Normalized Mutual Information (NMI) between the two partitions [36]. The NMI is bounded between 0 and 1 with where two partitions which match perfectly will have NMI equal to 1 and partitions which disagree tend to 0.

4.1 *Network creation*

The networks are created by measuring correlations between the activity of synthetic neurons (Fig. 2). The neural activity is generated with two goals in mind. First, we need to create a structure with state changes in order to test our method of updating interlayer edges. We are also motivated by brain activity in which neurons are sequentially recruited into a highly active and synchronized state, such as might occur in seizure dynamics. We would therefore like our network to model this type of spreading activity. To accomplish these goals, we generate a network in which a single community of highly active neurons sequentially merges with other low activity neurons.

Specifically, we generate 12 s of activity for 100 neurons via a Poisson process [37]. The Poisson process requires a firing rate parameter which controls the number of spikes per second a neuron will fire on average. The firing rate parameter for each neuron is initially set to 10 spikes/s. The network is created in such a way that each neuron undergoes a state change at which point the firing rate increases

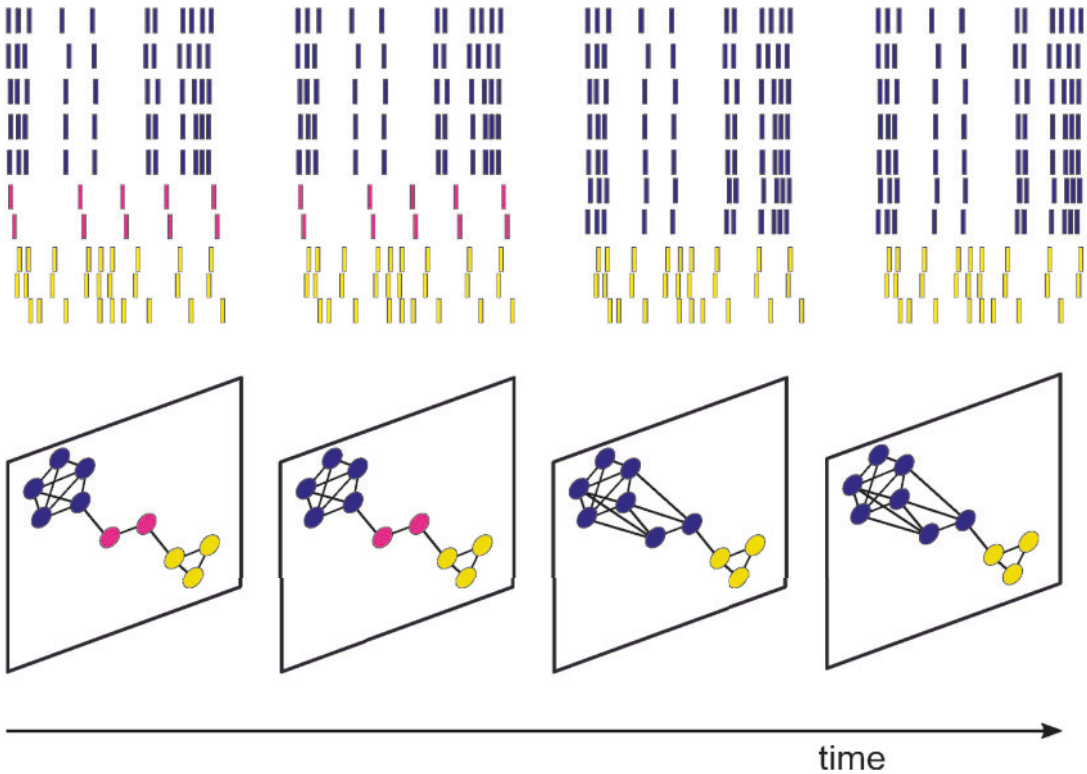

FIG. 2. Mapping spike trains to temporal networks. *Top*: Spike train data for 10 neurons. The rows correspond to the activity of the neurons, and a bar represents a time point at which the neuron was active or 'spiked'. *Bottom*: A temporal network created by measuring correlations in activity between spikes of neurons. Colour indicates community assignments based on similarity in firing activity.

to 30 spikes/s and the neuron synchronizes its firing pattern with all other neurons with this increased firing rate. In the first 1–2 s of activity, all neurons are generated with a firing rate of 10 spikes/s. In seconds 2–3, 10 of the 100 neurons undergo the state change (their firing rate parameter is increased and their activity becomes synchronized). This synchronization is achieved by generating a single master spike train and then using this train to generate correlated trains. The correlated trains are created by randomly jittering the positions of spikes (within a 5 ms window) and randomly deleting 10% of the spikes. In time 3–4 s, another 10 neurons undergo a state change and all synchronize with the previous 10 synchronized neurons, forming a group of 20 neurons with an increased firing rate and synchronous activity. This process continues until seconds 10–12 where all neurons are synchronized and remain this way. As commonly done when analysing spike train data [37], we then convolve each spike with a Gaussian with a standard deviation of 1.2 ms. We build a functional network by measuring the absolute value of the pairwise Pearson correlations between the neural activity of each neuron in a given time window. The choice of time window will affect the correlations measured within that time window and can have an impact on the results of community detection [38]. To account for this, we choose three time windows of size 1, 1.5 and 2 seconds, respectively. These choices represent different levels of mismatch between the window and the underlying structure. We call the 1 s window 'matching', the 1.5 s window

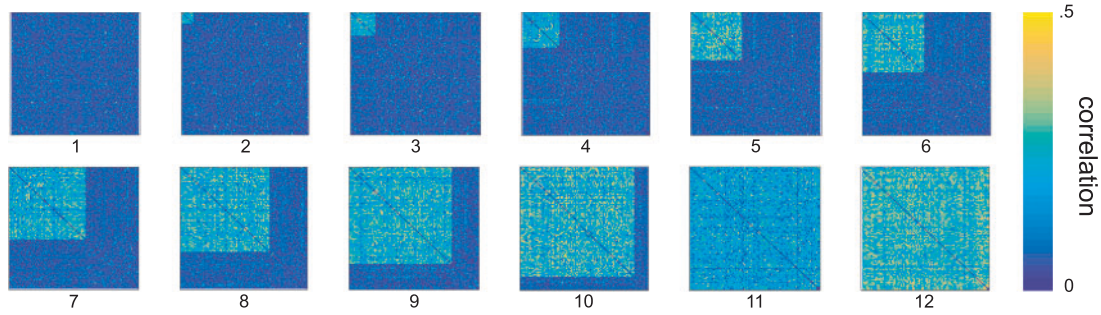

FIG. 3. Temporal network structure. The adjacency matrices for each layer of a sample network with matching window size (1 s). The i,j value of each matrix is the absolute value of the Pearson correlation between the activity of neuron i and neuron j in the given time window. Notice that at each layer, 10 neurons join the growing synchronized community. This network structure represents a sequence of mergers of communities.

'disjoint' and the 2 s window 'large' to indicate the level at which the window overlaps the dynamic structure of changes within the network. See Fig. 3 for an example of the intralayer adjacency matrices for the matching window case.

4.2 *Parameter sweep and null networks*

To assess the performance of our method on the simulated network, we attempt to find near optimal parameters (γ and ω) with which to compare our results. To do this, we choose (γ, ω) in $[0, 2] \times [0, 2]$ discretized by a step size of 0.05. This gives 1681 points in in the parameter plane, and for each of these points, we run a fixed consensus over 100 runs of the modularity maximization algorithm. We do this for each of the 100 sample networks so that, for each sample, we can determine the parameters that give the highest NMI when using a fixed consensus (optimal parameters). The parameters found this way are optimal in the sense that they maximize the NMI with respect to the ground truth and thus represent the best choice of parameters.

Note that in practice, doing such a sweep to find optimal parameters is not possible since one cannot compare the output partition to the ground truth. In addition, performing a fixed consensus (100 runs of the heuristic) over 1681 possible pairs of parameters is computationally expensive. Thus, while we perform this task here for comparison, performing a parameter sweep to find the optimal parameters is not a feasible option.

In Fig. 4, we show the mean NMI over all 100 samples for each choice of parameter value and for each of the two null networks in consideration. Notice that when using the NG null network, there are two phases of the parameter plane separated by $\gamma \sim 0.9$ and that, in general, the NMI for the NG network is lower than that of the U network. There are exceptions to this behaviour, such as can be seen in the 'disjoint' case where for a range of parameters the NG network performs better. However, this is due to the properties of the NMI measure used to quantify performance. In this case, both the U and NG null networks incorrectly classify the community structure: when using the U null network, many nodes are incorrectly grouped into a single community across all layers. However, when using the NG null network, the nodes are broken apart across all layers. The NMI is naturally biased towards the second classification, resulting in slightly higher values of the measure when using the NG network, despite the poor performance of both null networks.

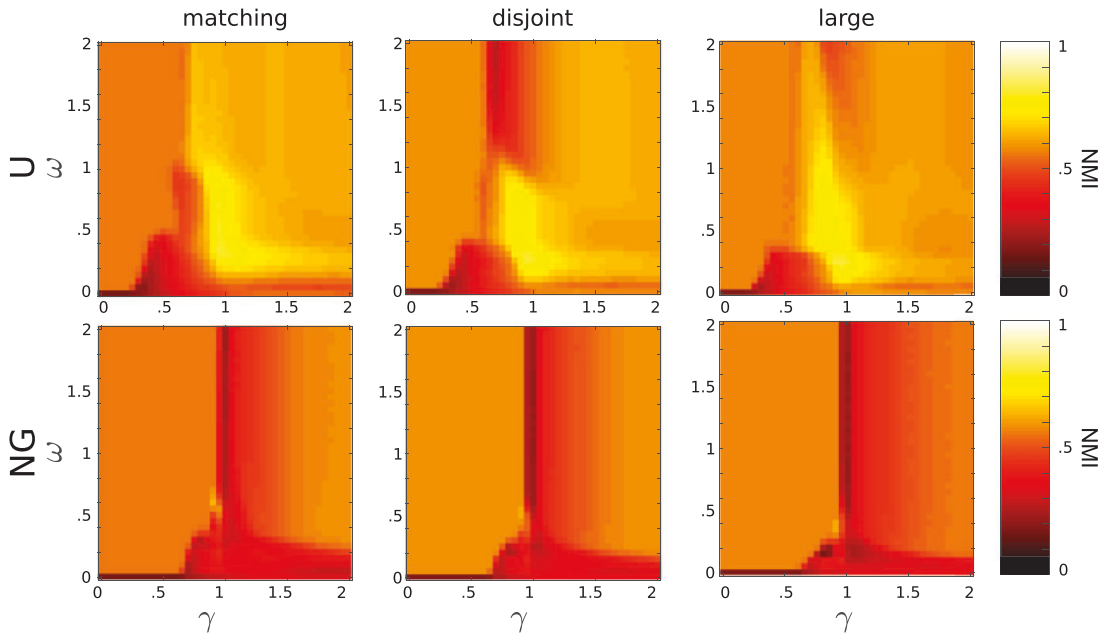

FIG. 4. Modularity parameter landscape. The mean NMI over all 100 samples for each choice of parameters. *Top*: The mean NMI using the uniform null network for each window choice and for $\omega \in [0, 2]$ and $\gamma \in [0, 2]$. *Bottom*: The mean NMI using the NG null network for each window choice and for $\omega \in [0, 2]$ and $\gamma \in [0, 2]$. Notice that the uniform null network has a region of high NMI for $0.2 < \omega < 1$ and $\gamma \sim 1$ whereas the NG null network exhibits a sharp phase transition in when $\gamma \sim 1$.

Although the exact values of optimal parameters varies within the 100 sample networks, as seen in Fig. 4, these values are approximately within the range $\omega \in [0.3, 1]$ and $\gamma \in [0.9, 1.3]$ for the U null network and $\omega \in (0, 2], \gamma \in [0, 0.9]$ for the NG network.

4.3 *Comparing community detection parameters*

Having found the optimal parameters for each of the 100 sample networks over all 3 time windows and each of the 2 null networks (U, NG), we can compare our method with the performance of the algorithm with these optimal parameters. We run a fixed consensus with optimal parameters both with and without updates, and we also run a sweep consensus with and without updates.

To perform updates, we first measure the firing rate for each neuron in each time window. For a fixed pair of consecutive time windows, t_1 and t_2, for each neuron, we compute the difference in firing rate between these windows. If the difference in firing rates is $\tau = 2$ or more standard deviations above or below the mean, we update the interlayer edge value between windows t_1 and t_2 according to the procedure in Section 3.2. Thus, our notion of similarity is based on population statistics: if a neuron's change in firing rate is significant relative to the population, we update its interlayer edge to reflect the decreased self-similarity between time layers. Here, we also calculate a 'meta mean' equal to mean of the distribution of means across layers and a 'meta standard deviation' equal to the standard deviation of the standard deviations of changes across layers. We do not apply updates to any nodes such that the standard deviation of change between layers is less than the meta mean minus two times the meta standard deviation. While this control is meant to exclude performing updates in layers that do not exhibit change,

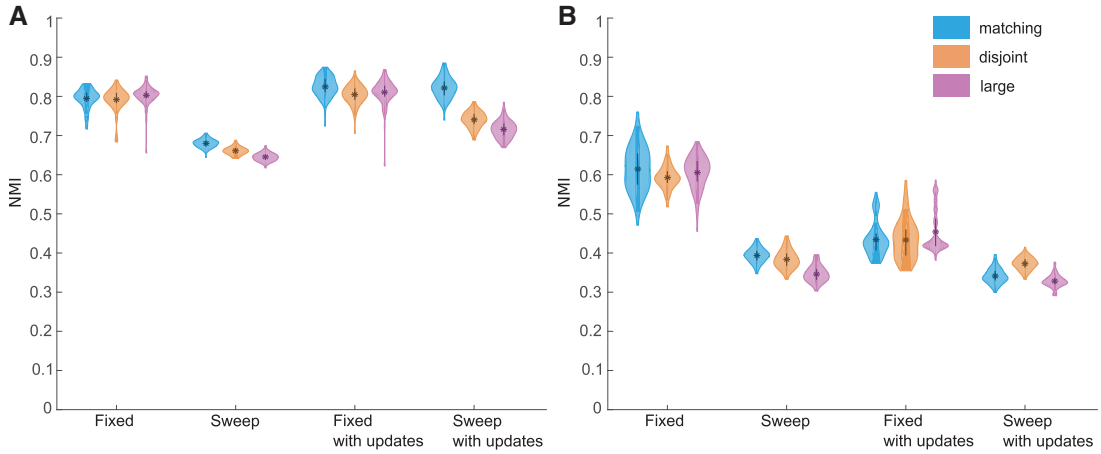

FIG. 5. NMI distributions. The normalized mutual information (NMI) over the 100 samples for each time window and each method of parameter selection. In the fixed method, γ and ω are selected to be the optimal parameters identified in Section 4.2 and no updates are applied; in the sweep method a consensus is run over $\omega \in [0, 2]$ and $\gamma \in [0, 2]$, and no updates are applied. We also run each of these methods when updates are applied using $\omega = \omega_{\text{global}}$ and $\rho = 0.1$. (A) Results for the U null network. Notice that a sweep with updates performs nearly as well as a consensus over an optimal choice of parameters. (B) Results for the NG null network. The NMI for the NG null network is fairly low across all methods.

in our simulated data, its effect on performance is minimal. To perform a sweep with updates, as before, we choose (γ, ω) in $[0, 2] \times [0, 2]$ discretized by a step size of 0.05, and set $\rho = 0.1$.

In Fig. 5, we show the distribution of NMI for each of the 300 networks (100 samples and 3 time windows) and for both null networks. It is immediately clear that the U null network outperforms the NG network on all fronts. Notice that for the U null network, the sweep consensus with updates performs comparably to the fixed consensus with and without updates. Intuitively, one would expect the optimal parameters to significantly outperform the sweep consensus since the comparison is between parameters found to be optimal with an indiscriminate range of parameters. Surprisingly, the sweep consensus together with updating interlayer edge weights gives similar performance to that with optimal parameters, indicating it may be a promising tool in practice when optimal parameters are not available.

To further quantify the performance of these methods, we show the dynamic community structure found via each method in Figs 6 and 7. Since we have 100 sample networks, we only choose a single representative result which is chosen so that the NMI of the representative is close to the average NMI over all 100 samples. That is, the displayed community structure represents the average community detection performance. We colour the larger growing community in pink for better colour differentiation.

It should be made clear that the NMI, like all measures of similarity between partitions, is not perfect. For example, it has been shown that the NMI favours structures with greater number of communities [39–41]. Although the NMI distributions are drastically lower for the NG null network, Figs 6 and 7 provide indications of what drives the differences. The NG null network tends to group the uncorrelated nodes into one large community where as the U null network does a better job of separating them. This is likely driving the majority of the difference in the NMI between the two choices of null networks. For the fixed method with the NG network, one can clearly see the multilayer resolution limit takes effect in the first and last time layers, where small community mergers cannot be detected. With the U null network, the resolution limit is not as drastic and is not present in the last layers of the network. This is what

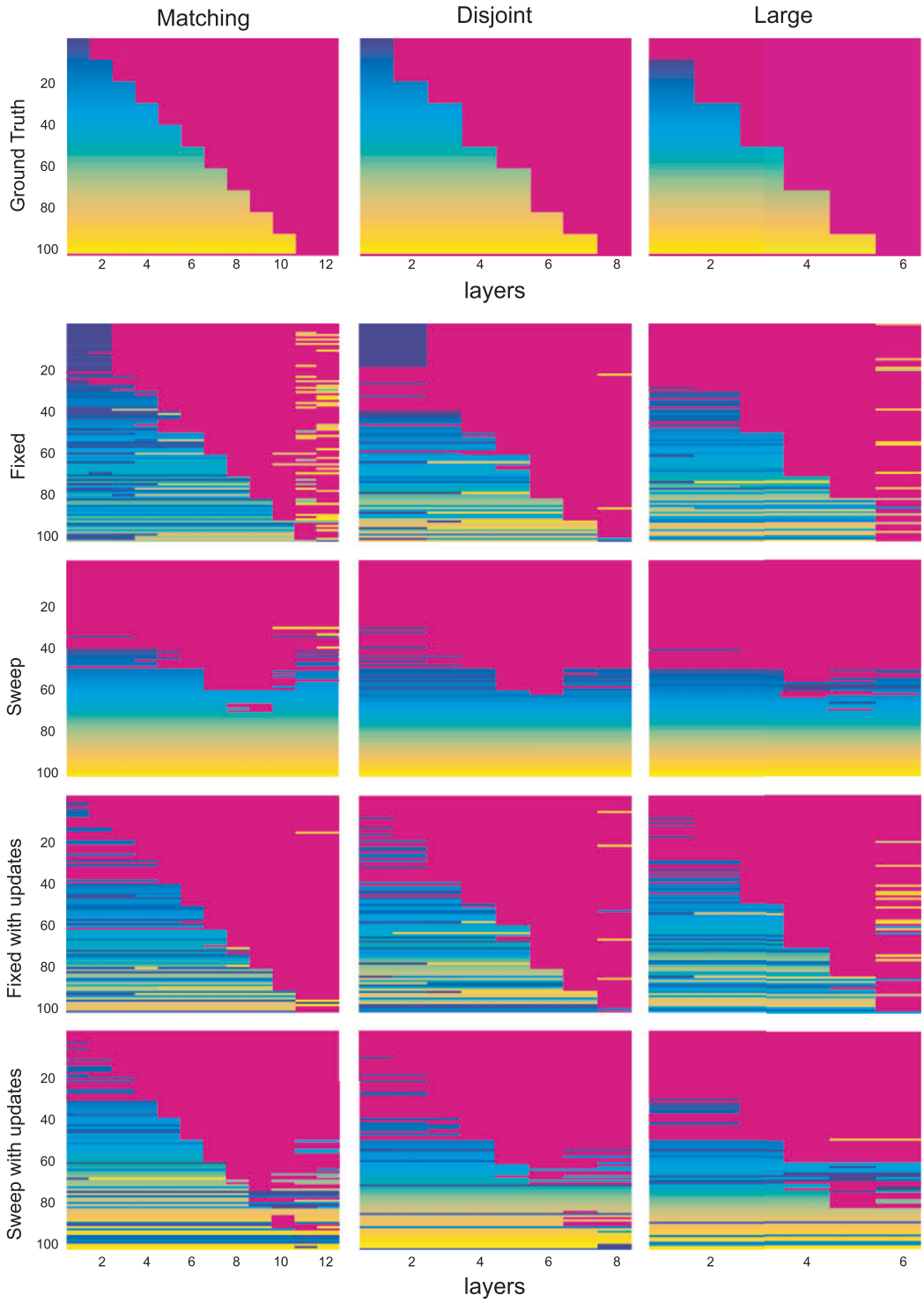

FIG. 6. Average community detection for the U null network. Representative partitions found by the U null network for each method and time window. The representative is chosen so that the NMI of the representative with the ground truth is close to the average NMI taken over all 100 samples. When compared with Fig. 7, we find that the use of the U null network results in superior separation of uncorrelated nodes.

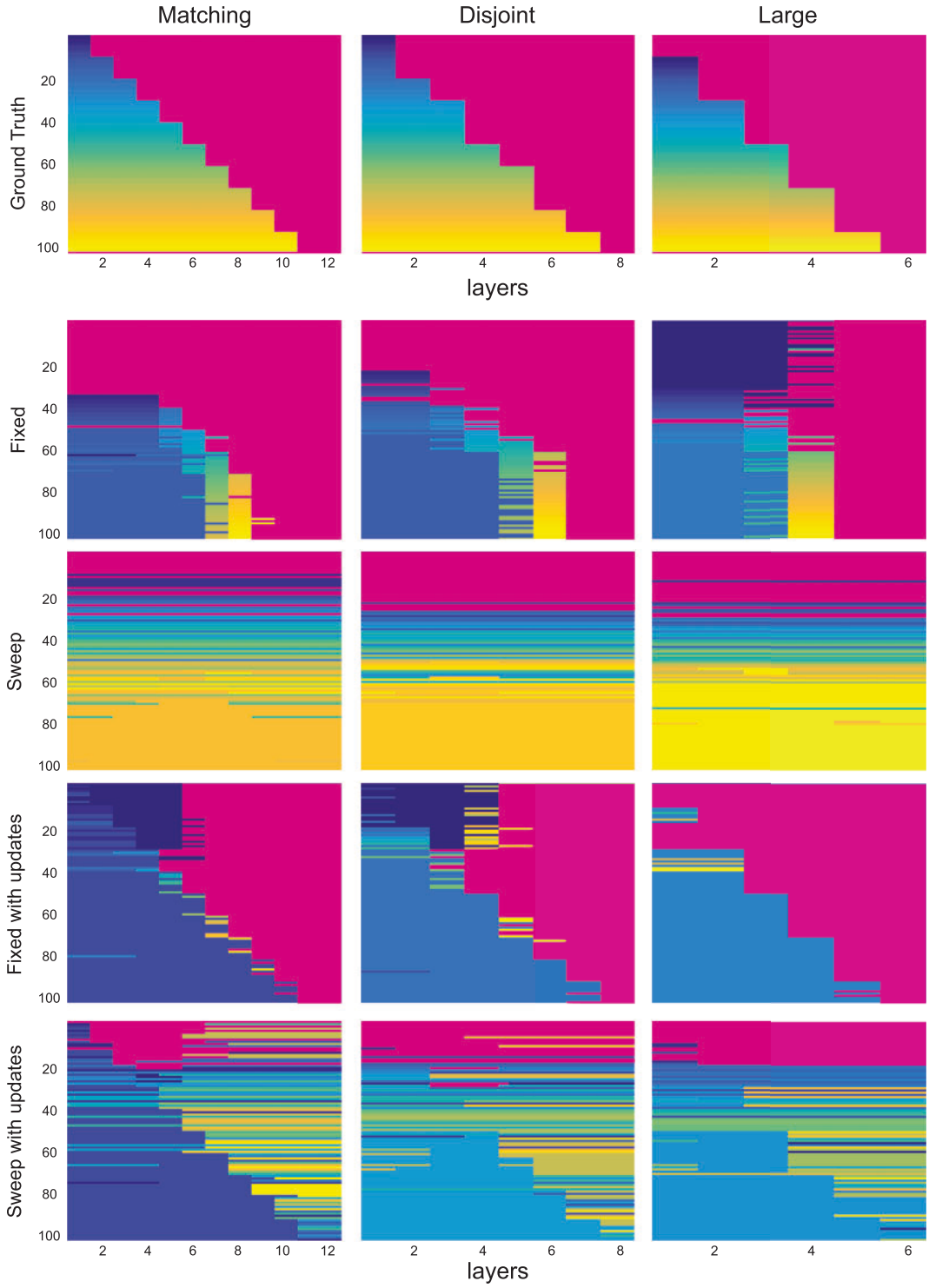

FIG. 7. Average community detection for the NG null network. Representative partitions found by the NG null network for each method and time window. The representative is chosen so that the NMI of the representative with the ground truth is close to the average NMI taken over all 100 samples. We see that the uncorrelated nodes tend to be grouped into one large community, which likely explains much of the poorer performance compared to when using the U null network.

we expect from the formula for Ω_U given in Equation 4. For example, we can compute Ω_{NG}, and find that it is $\Omega_{NG} = 0.0013$ and $\Omega_{NG} = 0.0157$ for the last final layer containing a merger in the matching window case. Comparatively, for the U network, in the first layer $\Omega_U = 0.0046$, whereas in the last layer $\Omega_U = 1.0296$. Finally, we notice that for the NG null network and the sweep with updates method, the uncorrelated nodes are grouped into a single community while the larger community of correlated nodes are separated. This is the exact opposite of the ground truth. This phenomenon is not present in the U null network, which gives very good performance with the sweep consensus with updates.

One can also ask how performance changes for different values of the percentage of change parameter, ρ, or the parameter, τ, that sets the number of standard deviations above or below the mean a change in nodal properties must be to be selected for updates. In Fig. 8, we show the NMI as a function of ρ, when $\tau = 2$, and as a function of τ when $\rho = 0.1$ for both the U and NG null networks. For the case of the U null network, we see improved performance with decreasing ρ and τ, indicating that larger changes to more nodes results in better performance. For the NG network, we do not see the same trend, likely due to the overall poor performance of the method using this null network (note the difference in range of scale of NMI between panels A/C and B/D).

In the current study, we were only able to measure optimal parameters because we had access to the ground truth. However, our main result suggests that using the U null network with a sweep consensus and updates gives nearly optimal performance. In most real-world settings, the ground truth is not known and one will not have access to the optimal parameters, but one may have access to a self-similarity measure for the nodes. In this case, a sweep consensus together with interlayer updates would be appropriate. We explore this option in the following section.

5. Applications to experimental data: calcium imaging of neuronal state changes

We now examine the performance of our methodology when applied to real-world experimental data monitoring the spiking activity of hundreds of neurons in the mouse sensorimotor cortex during the transition between awake and anaesthetized states. Neuronal activity was recorded using two-photon microscopy to monitor the calcium activity of individual neurons in head-fixed mice that were free to run on a spherical treadmill. For detailed experimental materials and methods, please see the Appendix.

We analysed data for two experimental subjects that entered deep anaesthesia and recovered from anaesthesia within minutes such as could be captured during a two-photon calcium imaging session. As depicted in the timelines in Figs 9 and 10, the isoflurane (anaesthesia) was introduced at approximately time = 6000 frames and turned off at approximately 11000 frames. The response of individual mice to isoflurane is variable and the star in each figure denotes the point at which the mouse entered the deeply anaesthetized state. One can see in the corresponding raster plots that depict the spiking activity of imaged neurons (right column of the figures) that the introduction of isoflurane induced synchronized firing and neuronal activity slowed when the mouse entered the deeply anaesthetized state.

We applied dynamic community detection both with and without updating interlayer edges to the spiking data of 129 neurons for each mouse, extracted from the calcium imaging data recorded during experiments. Unlike the simulated data in which we expected similar types of change in neuronal dynamics between layers, in this data set, we were uncertain where changes in neuronal firing rates might occur, and therefore chose to use the global updating rule based on the overall distribution of changes in firing rates across all layers. In Figs 9 and 10, we show the results of the community detection applied to the spiking data for each mouse when no updates are applied and ω is held constant across all layers, updates to interlayer edges are performed on the neurons between layers that exhibit the top 10% of changes in firing rates, and updates to interlayer edges are performed on the neurons between layers that exhibit

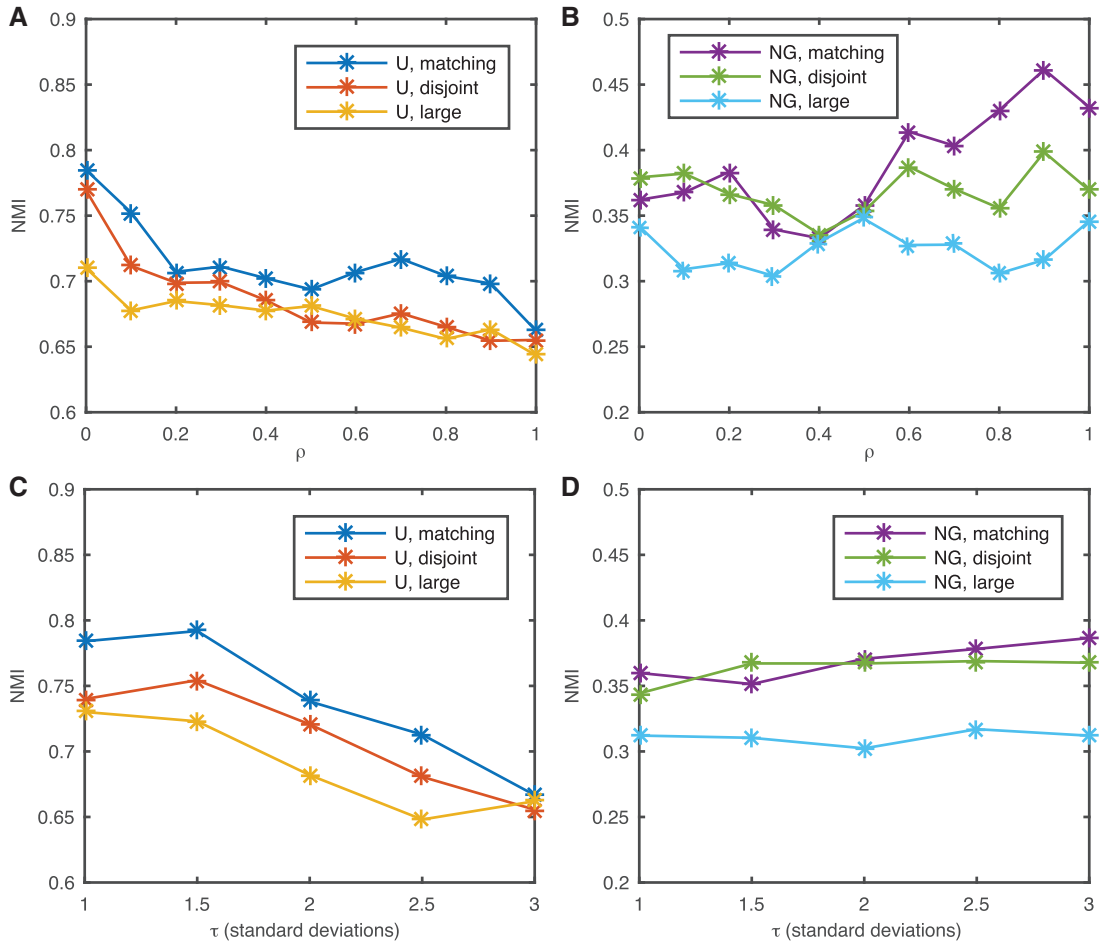

FIG. 8. Community detection for different updating thresholds. NMI quantifying the performance of community detection applied to the representative networks of Figs 6 and 7 for the three different window sizes. (A) Results using the U null network as a function of varying ρ while $\tau = 2$. (B) Results using the NG null network as a function of varying ρ while $\tau = 2$. (C) Results using the U null network as a function of varying τ while $\rho = 0.1$. (D) Results using the NG null network as a function of varying τ while $\rho = 0.1$. For the case of the U null network, we see improved performance as greater changes are applied to more nodes.

the top 30% of changes in firing rates. In all cases, we used a window size of 1000 frames to calculate firing rates, a value of $\rho = 0.1$ to perform updates to interlayer edges, and the U null network. Here, the colour of each community has been additionally mapped to the spiking activity of the neurons in order to more easily visualize the evolution of communities as neuronal dynamics change. Note that because the plots are sorted by community assignment, the neuron ID number does not necessarily represent the same physical neuron in each row of analysis (no updates, 10% updates and 30% updates).

In the experimental data analysed here, it is impossible to know the expected evolution of community structure, and we present this analysis only as an example of how applying interlayer updates to real-world data can produce different patterns of community evolution. In this data, we indeed see clear differences in the patterns of community structure when interlayer edges are not updated versus when updates are

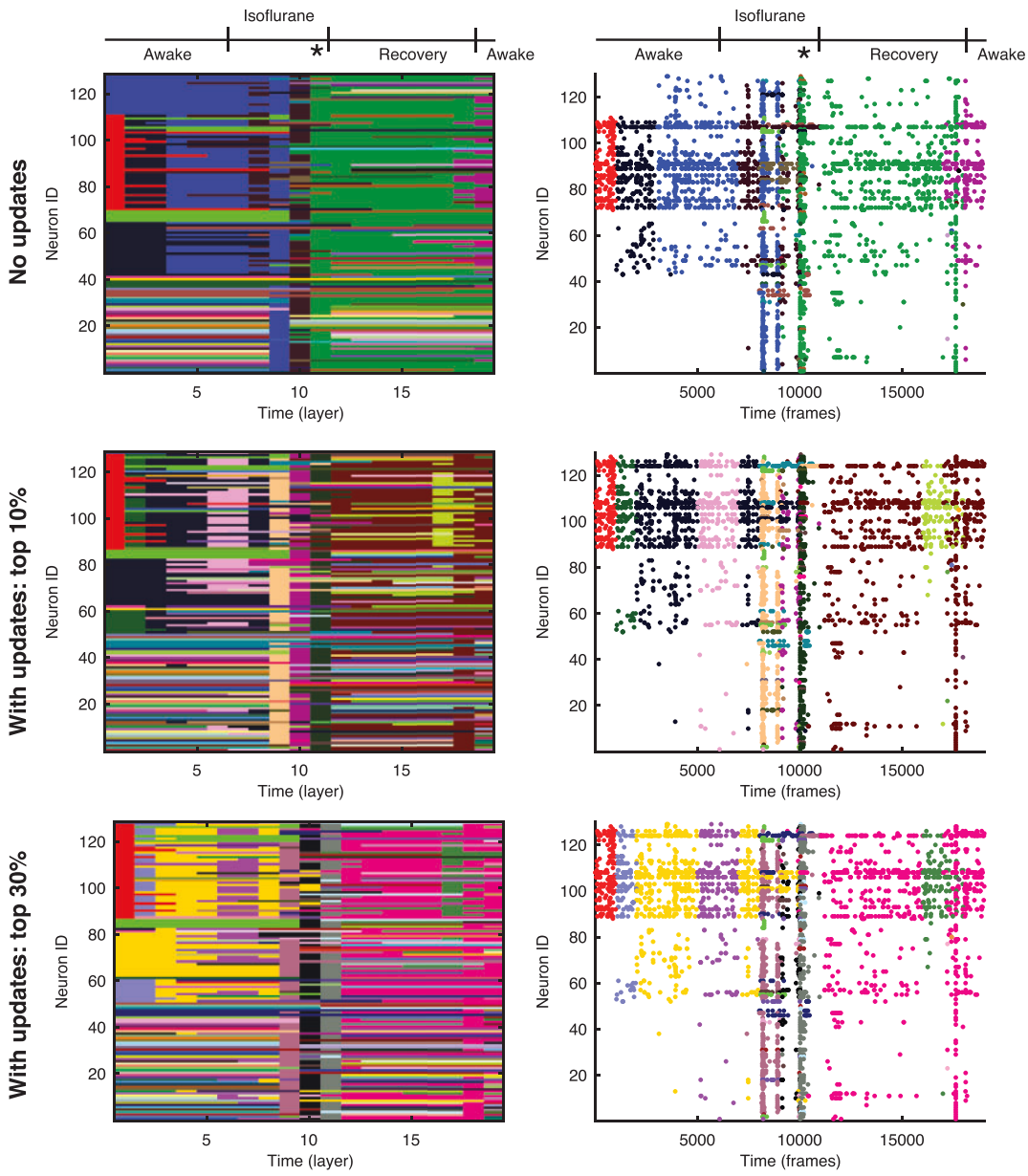

FIG. 9. Community evolution for mouse 1. Evolution of detected community structure for mouse 1 during the transition between awake and anaesthetized states. The experimental timeline is depicted at the top and the star denotes the point at which the animal enters the deeply anaesthetized state. (*Left*) The evolution of community assignment (depicted by colour) for 129 neurons based on three different types of analysis: no updates of interlayer edges (top), updates applied to the top 10% of changes in firing rates between layers (middle), and updates applied to the top 30% of changes in firing rates between layers (bottom). (*Right*) The corresponding spiking activity of the same 129 neurons, colour coded by community assignment at the time of each spike. We see clear differences in the evolution of community structure when updates are applied as to when they are not.

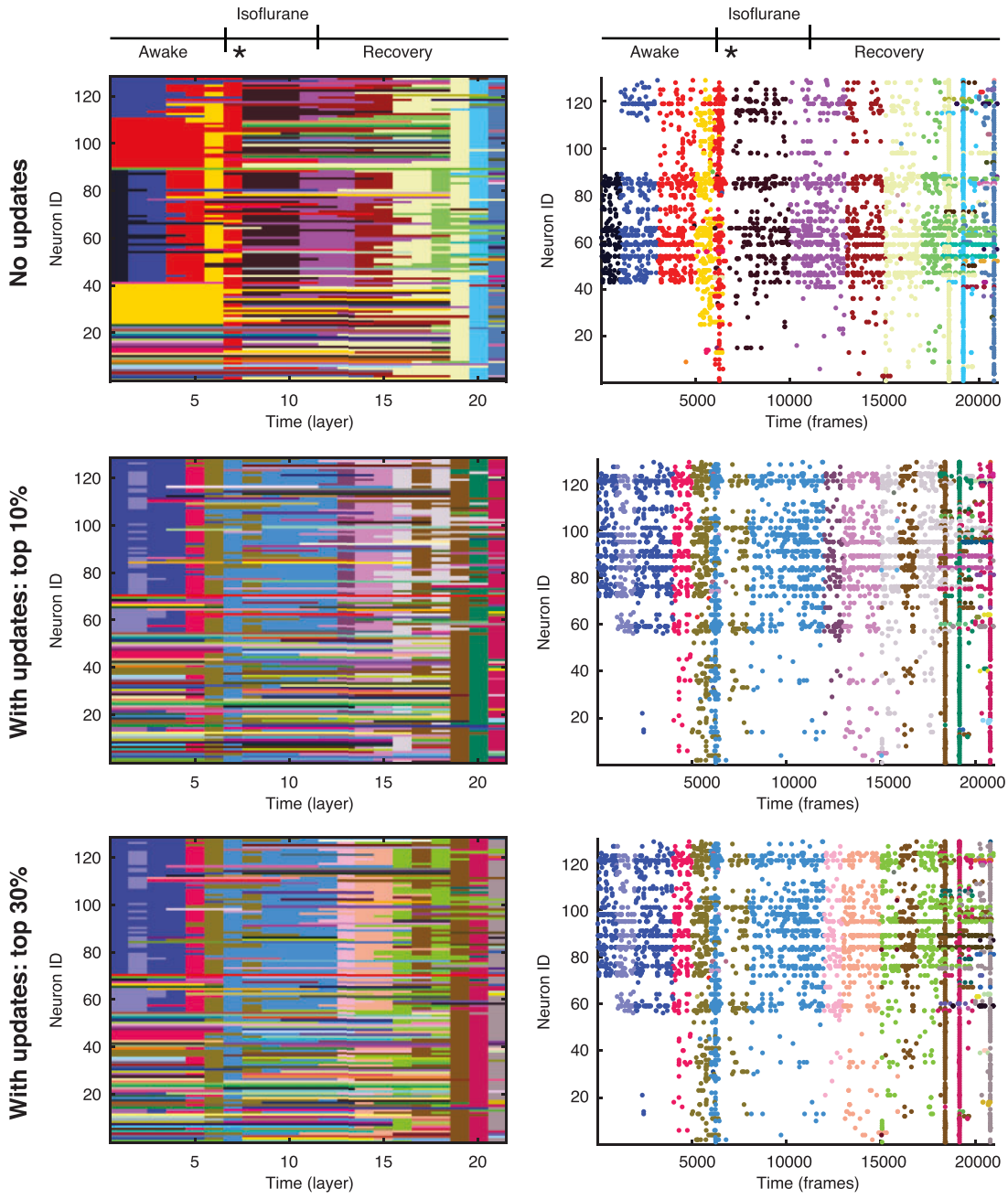

FIG. 10. Community evolution for mouse 2. Evolution of detected community structure for mouse 2 during the transition between awake and anaesthetized states. The experimental timeline is depicted at the top and the star denotes the point at which the animal enters the deeply anaesthetized state. (*Left*) The evolution of community assignment (depicted by colour) for 129 neurons based on three different types of analysis: no updates of interlayer edges (top), updates applied to the top 10% of changes in firing rates between layers (middle), and updates applied to the top 30% of changes in firing rates between layers (bottom). (*Right*) The corresponding spiking activity of the same 129 neurons, colour coded by community assignment at the time of each spike. We again see clear differences in the evolution of community structure when updates are applied as to when they are not.

applied. The differences in community evolution when the number of edges that are selected for updates is varied are more subtle; overall the community evolution looks similar when updating either the top 10% or top 30% of the possible changes. One clear difference between the detected community structures is the presence of more single neuron communities being detected when updates are applied. When updates are not applied, we see more large communities that are composed of non-firing neurons. This result is a consequence of the resolution limit. When building the functional networks, we made the choice to define non-spiking neurons to be unconnected to other neurons within the layer and therefore would like to detect these neurons as single-neuron communities in the analysis. It is therefore clear that applying updates to interlayer edges improves the community detection performance in this regard.

Understanding how the differences in evolution of community structure relate to the changes in the state of the mouse is more complex, and we can only speculate as to what different patterns of evolution might depict. In Fig. 9, we see that mouse 1 does not appear to have a change in its activity patterns when the isoflurane is first introduced, and therefore it is somewhat unsurprising that no method of analysis detects a change associated with the introduction of the anaesthesia. However, we do see that the point of deep anaesthesia (depicted by the star in the figure) is associated with the recovery period when no updates are applied and is instead detected as its own separate state in both cases when updates are applied.

In mouse 2 (Fig. 10), we see a different response to the introduction of anaesthesia, and the mouse becomes deeply anaesthetized almost immediately upon introduction of the isoflurane. Here, when updates are applied when analyzing the data, we see the emergence of a blue community that somewhat corresponds to the time when isoflurane is applied, and there is an initial switching between green and blue communities that could be associated with the transition into the anaesthetized state. Further, the application of updates results in more switching of communities (states) during the recovery period than when no updates are applied in the analysis. Because the exact effects of anaesthesia are not well understood, it is difficult to understand how this relates to the recovery of the mouse, and multiple further experiments that are beyond the scope of this article would need to be performed in order to map the detected communities to possible states associated with the recovery of the animal. However, the fact that applying self-similarity based updates to the community detection procedure results in modified patterns of the evolution of community structure suggests that the method is indeed useful when analysing real-world data.

6. Discussion and conclusions

Functional temporal networks are an important class of networks for modelling dynamic nodes, and the community structure in these networks can evolve through the temporal layers. It is therefore important to develop methods that are sensitive to these community changes, especially when the system undergoes a state change. In this article, we addressed this issue with regards to the popular modularity maximization method for dynamic community detection. Our contribution is three-fold: we (i) gave a theoretical result on the difference in choice of two popular null modals for modularity maximization in functional temporal networks; (ii) introduced a method for setting interlayer coupling on a layer by layer basis; and (iii) showed how a minimal parameter consensus together with our method provides a robust method of community detection when applied to simulated neural data that exhibits state changes.

Although when using multilayer networks to model physical systems it is commonly assumed that interlayer coupling is constant, this leads to an inflexible model that does not necessarily reflect the dynamics of the system. Here, we have proposed a method that updates interlayer coupling values based on properties of nodal self-similarity between layers. The method is flexible and allows the user to choose

an application-dependent criterion for determining how and/or when to update interlayer links. However, this flexibility also introduces uncertainty in how to choose the criterion for updating, and we therefore have tried to provide guidance as to how one might think about making such choices. In our simulated data set, we knew that similar types of changes would be occurring in-between layers and therefore chose to updates nodes using a local threshold based on the standard deviation of the change in nodal properties between layers. However, in our experimental data set where we were uncertain when changes might occur, we instead used a global threshold based on the pooled distribution of nodal changes across all layers.

The value of the threshold as well as the extent to which edges are modified can also affect the performance of the method. We found that, in our simulated data, when using the U null network, changing the value of ρ that determines how much edges are updated showed improved performance for more drastic changes (low values of ρ), and that performance also improved as more edges were selected to be updated. This suggests that future work might explore the possibility of updating all edges using a value of ρ that is proportional to the level of self-similarity between layers. Our observation that the representation of interlayer links should be reflective of system states over time could of course be extended further, and all values of interlayer coupling could be derived based on experimental measures of nodal similarity over time. See Aslek *et al.* [42] for an example of defining interlayer links based on local measurements of information flow. We hope that the findings presented here will drive future work exploring more complicated and realistic methods for setting interlayer coupling values. However, our results do suggest that even our simple method of updating interlayer coupling values provides increased sensitivity to the detection of state changes in functional temporal networks, and we encourage others to explore the use of this method in a wider array of dynamic networks where state changes are observed.

Funding

This work was supported by the National Science Foundation (SMA-1734795 to S.F.M. and SMA-1734813 to E.M.G.); the National Health Institute (K08 NS097633 to E.M.G.), a Burroughs Wellcome Fund Career Award for Medical Scientists to E.M.G.; and by a University at Buffalo Computational and Data-Enabled Science and Engineering Fellowship to M.V. The content is solely the responsibility of the authors and does not necessarily represent the official views of any of the funding agencies.

A. Detailed experimental materials and methods

A.1 *Experimental animals*

All animal experiments were performed using mice in accordance with The Children's Hospital of Philadelphia Animal Care and Use Committee. Animal holding rooms were maintained at 22°C on a 12 h light/dark cycle. We used both male and female mice given the mandate of the US National Institutes of Health (NIH) to include both sexes in all preclinical studies [43].

Mouse strains used included wild-type C57BL/6J mice (Jackson Laboratories Stock No. 000664; RRID:IMSR_JAX:000664), PV-Cre (Jackson Laboratories Stock No. 033863; RRID:IMSR_JAX:008069) and tdTomato (tdT) reporter strain (Jackson Laboratories Stock No. 007914; RRID:IMSR_JAX:007914). PV-Cre and tdT reporter mice were maintained as homozygotes; PV-Cre mice were then bred to tdTomato reporter mice to generate double-heterozygous progeny on a 100% C57BL/6J background in which all mice express tdTomato in PV cells.

All efforts were made to minimize pain, discomfort, and distress by using anaesthesia and analgesia for all surgical procedures and via detailed post-operative monitoring and care. For surgery, animals are

deeply anaesthetized by inhalation of isoflurane in oxygen (4% for induction, 1–2% for maintenance, in 0.8 L/min oxygen). Buprenoprhine-SR (0.5–1.0 mg/kg) is used to reduce stress and surgical pain, with additional doses given as indicated. Depth of anaesthesia is evidenced by lack of spontaneous movement or vocalization and lack of response to tail or toe pinch.

A.2 *Injection of adeno-associated viruses (AAV) and preparation of chronic cranial window implantation*

The injection of AAV constructs or genetically encoded calcium indicators is widely used in neuroscience and is a standard procedure [44, 45]; this procedure is combined with placement of a chronic cranial window for *in vivo* two-photon calcium imaging [46]. Briefly, the anaesthetized animal is maintained in a stereotaxic frame, and its head is shaved and scalp sterilized. A skin incision is made along the midline using scissors and skull surface landmarks are identified with the target area (primary motor cortex) localized using stereotaxic coordinates. A small (3 mm) craniotomy is made and a flap of bone removed to expose the dura. Viral vectors, typically 20–100 nL of $2.5–5 \times 10^{12}$ GC/mL solution, are then injected using a beveled glass micropipette positioned in layer 2/3 neocortex at a rate of 10–20 nL/min. A round 5 mm diameter cover slip is glued to the surface of the skull overlying the craniotomy and a custom-made titanium restraint bar is cemented to the skull surrounding this cover glass. The optical window and restraint bar are sealed to the skull with dental cement.

A.3 *Behavioural habituation for in vivo imaging*

Mice can be easily trained to tolerate prolonged head fixation using mild water restriction, which is widely used for habituating awake mice to tolerate head fixation for *in vivo* imaging and behaviour [47]. Mice are given ∼35% of daily ad libitum water consumption until weight drops by 15–30% of baseline. Mice are monitored daily for weight, activity level, posture and grooming, and signs of dehydration. We maintain a daily health assessment for these mice. If health scores increase to a pre-established threshold, animals are given additional water; otherwise, animals receive daily water adjusted to maintain a weight that is 15-30% below baseline. After acclimatization to handling and then to head fixation, mice receive periodic water rewards during imaging sessions.

A.4 *Two-photon imaging*

Experiments were performed using a customized two-photon imaging system (Bruker; Billerica, MA) equipped with a resonant scanner (Cambridge Technologies) and high-sensitivity gallium arsenide phosphate (GaAsP) detectors (Hamamatsu). Imaging was performed with a MaiTai DeepSee Ti:Sapphire pulsed infrared laser (SpectraPhysics) with a 16X 0.8 NA objective (Nikon). Mice were allowed to run or rest on a custom-made spherical treadmill. After obtaining data during a baseline imaging period, anaesthesia was induced using 4% isoflurane in 0.8 L/min oxygen. When deep anaesthesia was reached as assessed by absence of movement or vocalization in response to tail and toe pinch, the isoflurane was turned off and anaesthesia was lifted. Whole-field images were acquired at 30.4 frames per second at 512 $\times$ 512 pixel resolution with PrairieView software and converted to .tif stacks for analysis.

A.5 *Spike extraction*

The resultant .tif movies were first corrected for x, y-motion artefacts, and neuron locations are manually segmented using the ImageJ graphical user interface. The segmentation results in a list of regions of

interest (ROIs), each one enclosing a neuron in the image (i.e. cell contours). To extract the activity of an individual neuron, the average fluorescence of the neuron is computed over the entire area of the corresponding ROI for each frame of the movie. The raw values are then normalized by computing the change in the fluorescence from the baseline divided by the baseline value, such that each neuron's calcium activity is expressed by its relative change in fluorescence over time. These normalized traces were then deconvolved using the OASIS software package [48], resulting binary spike train data depicting the firing times of each neuron.

A.6 *Network creation*

Calcium spikes were first convolved with a Gaussian with a standard deviation of $\sigma = 2.8$ frames to allow for noise in the detection process. The calcium movies were then divided into windows of 1000 frames to create temporal layers, resulting in 19 layers of data for mouse 1 and 21 layers of data for mouse 2. Functional temporal networks were created by defining intralayer links between neurons to be the absolute value of the Pearson correlation between the convoluted spike trains, such that all intralayer connections were within [0,1]. If a neuron did not fire within a given window, we define the intralayer connection between this neuron and all others to be equal to zero.

References

1. Holme, P. & Saramaki, J. (2012) Temporal networks. *Phys. Rep.*, **519**, 97–125.
2. Kivela, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y. & Porter, M. A. (2014) Multilayer networks. *J. Complex Netw.*, **2**, 203–271.
3. Masuda, N. & Lambiotte, R. (2016) *A Guide to Temporal Networks*. London: World Scientific.
4. Bassett, D. S., Yang, M., Wymbs, N. F. & Grafton, S. T. (2015) Learning-induced autonomy of sensorimotor systems. *Nat. Neurosci.*, **18**, 744–751.
5. Valdano, E., Ferreri, L., Poletto, C. & Colizza, V. (2015) Analytical computation of the epidemic threshold on temporal networks. *Phys. Rev. X*, **5**, 021005.
6. Moinet, A., Starnini, M. & Pastor-Satorras, R. (2015) Burstiness and aging in social temporal networks. *Phys. Rev. Lett.*, **114**, 108701.
7. Li, A., Cornelius, S. P., Liu, Y.-Y., Wang, L. & Barabási, A.-L. (2017) The fundamental advantages of temporal networks. *Science*, **358**, 1042–1046.
8. Taylor, D., Myers, S. A., Clauset, A., Porter, M. A. & Mucha, P. J. (2017) Eigenvector-based centrality measures for temporal networks. *Multiscale Model. Simul.*, **15**(1), 537–574.
9. Friston, K. J. (1994) Functional and effective connectivity in neuroimaging: a synthesis. *Hum. Brain Mapping*, **2**, 56–78.
10. Feldt, S., Bonifazi, P. & Cossart, R. (2011) Dissecting functional connectivity of neuronal microcircuits: experimental and theoretical insights. *Trends Neurosci.*, **34**, 225–236.
11. Muldoon, S. F. (2018) Multilayer network modeling creates opportunities for novel network statistics. Comment on "Network science of biological systems at different scales: a review" by Gosak et al.. *Phys. Life Rev.*, **24**, 143–145.
12. Mucha, P. J., Richardson, T., Macon, K., Porter, M. A. & Onnela, J.-P. (2010) Community structure in time-dependent, multiscale, and multiplex networks. *Science*, **328**, 876–878.
13. De Domenico, M., Sasai, S. & Arenas, A. (2016) Mapping multiplex hubs in human functional brain networks. *Front. Neurosci.*, **10**, 326.
14. Bazzi, M., Porter, M. A., Williams, S., McDonald, M., Fenn, D. J. & Howison, S. D. (2016) Community detection in temporal multilayer networks, with an application to correlation networks. *Multiscale Model.& Simul.*, **14**, 1–41.
15. Fortunato, S. (2010) Community detection in graphs. *Phys. Rep.*, **486**, 75–174.

16. Mucha, P. J. & Porter, M. A. (2010) Communities in multislice voting networks. *Chaos*, **20**, 041108.
17. Vaiana, M. & Muldoon, S. F. (2018) Multilayer brain networks. *J. Nonlinear Sci.*, 1–23, https://doi.org/10.1007/s00332-017-9436-8.
18. Newman, M. E. J. & Girvan, M. (2004) Finding and evaluating community structure in networks. *Phys. Rev. E*, **69**, 026113.
19. Newman, M. E. J. (2006) Modularity and community structure in networks. *Proc. Natl. Acad. Sci. USA*, **103**, 8577–8582.
20. Lee, S. H., Magallanes, J. M. & Porter, M. A. (2016) Time-dependent community structure in legislation cosponsorship networks in the Congress of the Republic of Peru. *J. Complex Netw.*, **5**, 127–144.
21. Yang, Z., Algesheimer, R. & Tessone, C. J. (2016) A comparative analysis of community detection algorithms on artificial networks. *Sci. Rep.*, **6**, 30750.
22. Weir, W. H., Emmons, S., Gibson, R., Taylor, D. & Mucha, P. J. (2017) Post-processing partitions to identify domains of modularity optimization. *Algorithms*, **10**, 93.
23. Muldoon, S. F., Costantini, J., Webber, W., Lesser, R. & Bassett, D. S. (2018) Locally stable brain states predict suppression of epileptic activity by enhanced cognitive effort. *NeuroImage: Clin.*, **18**, 599–607.
24. Bassett, D. S., Porter, M. A., Wymbs, N. F., Grafton, S. T., Carlson, J. M. & Mucha, P. J. (2013) Robust detection of dynamic community structure in networks. *Chaos*, **23**, 013142.
25. Pamfil, A. R., Howison, S. D., Lambiotte, R. & Porter, M. A. (2018) Relating modularity maximization and stochastic block models in multilayer networks. arXiv:1804.01964v1 [cs.SI].
26. Vaiana, M. & Muldoon, S. (2018) Resolution limits for detecting community changes in multilayer networks. arXiv:1803.03597v1 [physics.soc-ph].
27. MacMahon, M. & Garlaschelli, D. (2015) Community detection for correlation matrices. *Phys. Rev. X*, **5**, 1022–1034.
28. Gómez, S., Jensen, P. & Arenas, A. (2009) Analysis of community structure in networks of correlated data. *Phys. Rev. E*, **80**, 583–585.
29. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. (2008) Fast unfolding of communities in large networks. *J. Stat. Mech.*, **2008**, P10008.
30. Jeub, L. G. S., Bazzi, M., Jutla, I. S. & Mucha, P. J. (2011–2017) A generalized Louvain method for community detection implemented in MATLAB. http://netwiki.amath.unc.edu/GenLouvain (accessed October 25, 2017).
31. Good, B. H., de Montjoye, Y.-A. & Clauset, A. (2010) Performance of modularity maximization in practical contexts. *Phys. Rev. E*, **81**, 046106.
32. Lancichinetti, A. & Fortunato, S. (2012) Consensus clustering in complex networks. *Sci. Rep.*, **2**, 336.
33. Sarzynska, M., Leicht, E. A., Chowell, G. & Porter, M. A. (2015) Null models for community detection in spatially embedded, temporal networks. *J. Complex Netw.*, **4**, 363–406.
34. Jeub, L. G., Sporns, O. & Fortunato, S. (2018) Multiresolution consensus clustering in networks. *Sci. Rep.*, **8**, 3259.
35. Seifi, M., Junier, I., Rouquier, J.-B., Iskrov, S. & Guillaume, J.-L. (2013) Stable community cores in complex networks. In: (R. Menezes, A. Evsukoff & M. González eds) *Complex Networks*. Studies in Computational Intelligence, vol 424. Berlin, Heidelberg: Springer, pp. 87–98.
36. Danon, L., Diaz-Guilera, A., Duch, J. & Arenas, A. (2005) Comparing community structure identification. *J. Stat. Mech.*, **2005**, P09008.
37. Feldt, S., Waddell, J., Hetrick, V. L., Berke, J. D. & Zochowski, M. (2009) Functional clustering algorithm for the analysis of dynamic network data *Phys. Rev. E*, **79**, 056104.
38. Telesford, Q. K., Lynall, M.-E., Vettel, J., Miller, M. B., Grafton, S. T. & Bassett, D. S. (2016) Detection of functional brain network reconfiguration during task-driven cognitive states. *NeuroImage*, **142**, 198–210.
39. Vinh, N. X., Epps, J. & Bailey, J. (2010) Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.*, **11**, 2837–2854.

40. Amelio, A. & Pizzuti, C. (2015) Is normalized mutual information a fair measure for comparing community detection methods? *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015—ASONAM'15*, New York, NY, USA: ACM Press, pp. 1584–1585.

41. Jeub, L. G. S., Sporns, O. & Fortunato, S. (2018) Multiresolution consensus clustering in networks. *Sci. Rep.*, **8**, 1–16.

42. Aslak, U., Rosvall, M. & Lehmann, S. (2018) Constrained information flows in temporal networks reveal intermittent communities. *Phys. Rev. E*, **97**, 062312.

43. Clayton, J. A. & Collins, F. S. (2014) NIH to balance sex in cell and animal studies. *Nature*, **509**, 282–283.

44. Harris, J. A., Oh, S. W. & Zeng, H. (2012) Adeno-associated viral vectors for anterograde axonal tracing with fluorescent proteins in nontransgenic and cre driver mice. *Curr. Protoc. Neurosci.*, **59**, 1.20.1–1.20.18.

45. Cetin, A., Komai, S., Eliava, M., Seeburg, P. H. & Osten, P. (2006) Stereotaxic gene delivery in the rodent brain. *Nat. Protoc.*, **1**, 3166–3173.

46. Goldey, G. J., Roumis, D. K., Glickfeld, L. L., Kerlin, A. M., Reid, R. C., Bonin, V., Schafer, D. P. & Andermann, M. L. (2014) Removable cranial windows for long-term imaging in awake mice. *Nat. Protoc.*, **9**, 2515–2538.

47. Guo, Z. V., Hires, S. A., Li, N., O'Connor, D. H., Komiyama, T., Ophir, E., Huber, D., Bonardi, C., Morandell, K., Gutnisky, D., Peron, S., Xu, N.-l., Cox, J. & Svoboda, K. (2014) Procedures for behavioral experiments in head-fixed mice. *PLoS One*, **9**, e88678–16.

48. Friedrich, J., Zhou, P. & Paninski, L. (2017) Fast online deconvolution of calcium imaging data. *PLoS Comput. Biol.*, **13**, e1005423–26.