Linking through time: Memory-enhanced community discovery in temporal networks

Giulio Virginio Clemente*
IMT School for Advanced Studies, Piazza San Francesco 19, 55100 Lucca, Italy

Diego Garlaschelli
IMT School for Advanced Studies, Piazza S. Francesco 19, 55100 Lucca, Italy;
Lorentz Institute for Theoretical Physics, Niels Bohrweg 2, 2333 CA Leiden, The Netherlands;
and INdAM-GNAMPA, Istituto Nazionale di Alta Matematica, 00185 Rome, Italy

(Received 3 March 2024; accepted 4 October 2024; published 25 November 2024)

Temporal networks present a unique challenge regarding the community discovery task. The inherent dynamism of these systems requires an intricate understanding of memory effects and structural heterogeneity, which are often key drivers of network evolution. This study focuses on Markovian temporal networks and addresses these challenges with an innovative community detection method that introduces a modularity function. We specifically demonstrate how our approach enhances the detectability threshold, thereby improving the effectiveness of community detection in such a dynamic setting. We show that by associating memory directly with nodes' memberships and including it into the modularity expression, we can enhance the detectability threshold compared to scenarios where memory is ignored, thus extending the conditions under which communities can be accurately identified. We validate our approach through extensive numerical simulations, confirming its efficacy in a controlled environment. Additionally, by applying our method to real-world data, we not only demonstrate its practicality and robustness but also reveal its capacity to indirectly tackle additional challenges, such as determining the optimal time window for aggregating data in dynamic graphs.

DOI: 10.1103/PhysRevResearch.6.043204

I. INTRODUCTION

Models of real networks often describe complex systems as static entities. However, these systems dynamically evolve, with their connections undergoing continual changes. Such dynamism can significantly enrich network characterizations [1–3] and influence phenomena happening on them [4–8]. A key driver of this dynamism may be affiliation with specific mesostructures [9], which can thus identify and characterize relevant aspects of the system under examination. Particularly, the presence of communities or groups within these mesostructures has garnered significant interest within the scientific community. This focus is justified by the profound influence these communities can exert on various aspects of the system, such as the diffusion of processes within it [10], or its stability [11,12].

When considering the network's topology, communities emerge as clusters of nodes exhibiting a significantly higher link density than expected. Based on this aspect, numerous methodologies have been developed to tackle community detection, ranging from the use of objective functions to quantify the quality of a partition, such as the modularity function [13], to approaches based on generative models such as the stochastic block model [14], and flow models that leverage network dynamics to cluster nodes based on persistent flow patterns [15].

A fundamental concept on which this work is based is the definition of "expected," which is central to the modularity function, the main tool employed here. This function quantifies the deviation between a standard benchmark, which sets the expectations, and the observed values of specific measurements within the network, typically represented by links. Often termed a null model, this benchmark is essential for achieving accurate results, a significance we will discuss throughout this paper.

While the above-mentioned methodologies have been well developed for static networks, they encounter challenges when generalizing for time-varying graphs. Nevertheless, community detection for temporal networks is a rapidly expanding field, and several studies have endeavored to adapt static methods to the dynamic context [16–18]. The development of a robust modularity function for temporal networks meets a major obstacle due to the lack of suitable null models. These null models are crucial for incorporating time dependencies between nodes within the same community. If such dependencies exist and are not properly integrated into the benchmark, the performance of established algorithms can suffer [17]. Conversely, effectively incorporating time dependency can significantly enhance algorithmic performance and reveal new insights. In this paper, we fill this gap by

*Contact author: giulio.clemente@imtlucca.it

demonstrating how the maximum entropy approach [19] can be utilized to construct various forms of the modularity function and leverage them to account for different aspects of network structure. In our main result, we introduce a formulation of the modularity function based on a recent Markovian null model for temporal networks [20]. This formulation allows simultaneous control over anomalies related to link persistence between nodes and link formation, thus allowing an accurate generalization of the modularity function to time-varying graphs.

To demonstrate the advancements achieved by leveraging these properties, we investigate the problem of the detectability threshold, which serves as a theoretical limit defining the boundary for the ability of any algorithm to detect the presence of communities [21]. Initially, we introduce the threshold for the static case, and subsequently, by incorporating the insights presented in Ref. [22], we compute the threshold while considering the presence of memory between nodes within the same blocks. To support our theoretical results, we conduct a series of analyses on synthetic temporal networks, wherein the number of nodes and memberships remain fixed throughout the network's evolution while exploring various combinations of parameters. Our findings confirm theoretical expectations. Building on these results, we enhance the applicability of our modularity function by integrating a structural break detection method, which effectively captures significant changes in community structures. This adaptation improves our analysis of time-evolving communities and broadens the function's utility in dynamic network analysis.

In the final section of this study, we applied our proposed methodology to a real-world data set of proximity relations in a primary school [23], demonstrating its practical utility. We also addressed the challenge of selecting the optimal time window for constructing a temporal network. For our specific scenario, where we assume that memory is driven by nodes' membership, choosing a time window with heightened memory leads to a temporal network where community structures are more distinctly observable. We conclude this work with several key remarks and observations.

II. COMMUNITY DETECTION, NULL MODELS AND MODULARITY FUNCTION

Consider a graph $G = (N, \mathcal{E})$, where N represents a set of nodes and $\mathcal{E} \subseteq N \times N$ denotes the set of edges. A community structure within G is defined as a partition $\vec{g} = \{g_1, g_2, \ldots, g_q\}$ of N, where each $g_i \subset N$ is a cluster of nodes such that the density of edges within each g_i is significantly greater than the density of edges between g_i and $N \setminus g_i$. In this context, we focus on nonoverlapping communities, implying that each node belongs to exactly one g_i within $\vec{g}$. Identifying the optimal subdivision of a network with an inherent block structure is challenging and has attracted extensive research attention [24]. In response, Newman and Girvan introduced a quality function known as modularity in 2004 [13], designed to evaluate the effectiveness of network partitions. The original formulation of this function is expressed as follows:

$$Q(g) = \sum_{i,j \in N} [a_{ij} - \langle a_{ij} \rangle] \delta_{g(i)g(j)}, \tag{1}$$

where a_{ij} refers to the observed element of the adjacency matrix $\mathbf{A}$ for the graph G, marking the presence of a link between node i and node j. Conversely, $\langle a_{ij} \rangle$ is the expected value of the same link in a null model that presumes no community structure exists. The vector $\vec{g}$ specifically designates the community membership of each node.

The modularity function measures the deviation between the observed and expected number of intracommunity edges, as defined by the null model. The effectiveness of this measure lies significantly on the design of the null model, a pattern-generating model that maintains certain properties constant while allowing others to vary stochastically [25]. This randomization process is designed to simulate aspects of the data that would be expected in the absence of any specific mechanisms, such as community structure. It is precisely through this characteristic that the modularity function achieves its goal. Modularity is maximized with respect to the assigned node memberships to identify the optimal partition. When the optimal partition produces a modularity close to zero, it indicates a lack of strong community structure. Conversely, a modularity value approaching one signifies the presence of a strong community structure.

The choice of the null model is subjective and depends on the specific properties we aim to replicate in our random network. Significant focus has been placed on null models that accurately reproduce the degree sequence of undirected and unweighted networks, primarily to examine the impact of local inhomogeneity [26]. In this paper, we focus exclusively on null models that precisely reproduce the degree sequence of a given network. For binary and undirected networks, the degree-constrained Chung-Lu model is often used, defined by $\langle a_{ij} \rangle = \frac{k_i k_j}{2m}$, where k_i and k_j are the degrees of nodes i and j, respectively, and $2m$ represents the total number of possible links. However, while popular, the Chung-Lu model is an approximation valid only under stringent conditions [27], which many real networks do not meet. An example of this discrepancy can be seen in the work of Maslov *et al.* [28], who found that using the Chung-Lu model to calculate the connection probability between two hubs in an internet network snapshot on January 2, 2000, yielded a probability of 43.5, a clear anomalous value. Using such a flawed measure in the modularity function could introduce significant bias, adversely affecting the method's performance. To address these issues, we introduce a modularity function based on null models developed through a maximum entropy approach [19]. This method not only adheres more closely to the actual network structure but also avoids the biases inherent in models such as Chung-Lu.

A. Maximum entropy null models

The maximum entropy approach [29] is a methodology that enables the definition of a probability distribution over a set of elements, such as graphs, in a manner that ensures specific expected values are reproduced while all other aspects remain maximally random. Applying the maximum entropy formalism to real-world network-related problems involves a two-step process. The first step addresses a graph G with certain distinctive properties $C_i(G)$. In this step, we seek to obtain the most unbiased probability distribution $P(G)$ over

the set of all possible graphs, ensuring that the expected value of $C_i(G)$ aligns with the imposed constraints. To achieve this distribution, we leverage the Shannon-Gibbs entropy:

$$S[P] = -\sum_G P(G)\ln P(G). \tag{2}$$

Imposing that the probability distribution reproduces on average

$$\langle C_i\rangle = \sum_G C_i(G)P(G) = C_i^*, \tag{3}$$

where G is an ensemble element, C_i^* is the measured value of the constraint i, and $\langle\cdot\rangle$ indicates the expected value over the ensemble.

It is crucial to emphasize that, as we have defined it, this class of models allows for the imposition of constraints where the expected value, rather than the exact value for each network realization, is reproduced. This has implications not only on computational aspects [19] but also on the model's ability to replicate metrics not predetermined by constraints [30].

The constrained maximization problem is approached by employing the method of Lagrange multipliers ($\vec{\theta}$). The resulting probability distribution has the following functional form [31]:

$$P(G|\vec{\theta}) = \frac{e^{-H(G|\vec{\theta})}}{Z(\vec{\theta})}, \tag{4}$$

where $H(G|\vec{\theta}) = \sum_i \theta_i C_i$ is called graph Hamiltonian, and $Z(\vec{\theta}) = \sum_G e^{-H(G|\vec{\theta})}$ is the normalization constant.

The second step of this procedure involves determining the numeric values of the parameters $\vec{\theta}$ that reproduce the measured quantities on a real graph. To tackle this problem, we can employ the maximum likelihood principle, which entails setting the parameters to the specific values that maximize the likelihood function $P(G^*|\vec{\theta})$ given the model [32].

The first maximum entropy model we introduce is the binary configuration model [31], which serves as the maximum entropy counterpart to the Chung-Lu model. As previously noted, this model forms the basis for the classical definition of modularity. The binary configuration model (CM) is derived by imposing constraints on the degree sequence, resulting in the following Hamiltonian expression:

$$H(\mathbf{G}|\vec{\theta}) \equiv \sum_{i=1}^{N}\theta_i k_i, \tag{5}$$

where k_i is the degree of node i. Hence for this specific problem, we have that

$$\langle a_{ij}\rangle \equiv \frac{x_i x_j}{1+x_i x_j}, \tag{6}$$

where $x_i \equiv e^{-\theta_i}$ for each node i [19,32]. In the subsequent sections, we will employ maximum entropy null models to define new modularity expressions and characterize their properties.

B. Detectability threshold in static graphs

To explore the impact of temporal information on community detection, we investigate the behavior of various modularity functions in relation to the detectability threshold (DT) problem. This issue, which has attracted significant attention from researchers [14,33,34], describes a fundamental limitation in the ability of algorithms to discern community structures in networks, marked by the presence of a phase transition [34]. To frame this problem, we consider the well-known stochastic block model (SBM) [35], a generative model that produces networks with a block structure. This model serves as an ideal framework for examining the effectiveness of community detection algorithms under controlled conditions.

Given a set of nodes N and q groups denoted by $\vec{g} = \{g_1, g_2, \ldots, g_q\}$, the SBM in its basic formulation defines a network structure in which unweighted edges are placed between node i and node j with probability $p_{g_i g_j}$ depending on the groups which they belong. Hence the probability of observing a graph G is

$$P(G|g) = \prod_{(ij)\in N} p_{g_i g_j}^{a_{ij}}(1-p_{g_i g_j})^{1-a_{ij}}, \tag{7}$$

where a_{ij} is the element of the adjacency matrix $\mathbf{A}$ corresponding the graph G.

In the subsequent analysis, we focus on a specific class of block models where clusters are of uniform size, known as the planted partition model. All theoretical results in this work relate to this model configured with two communities. These communities are distinguished by a probability of connection between nodes of different clusters, p_{out}, and within nodes of the same cluster, p_{in}. For this specific case, it has been shown rigorously [36] that given a sparse graph G, i.e. with $p_{ij} = O(\frac{1}{N})$, with two communities, an average degree k, an average number of connections between groups $k_{\text{out}} = \frac{N}{2}p_{\text{out}}$, and within groups $k_{\text{in}} = \frac{N}{2}p_{\text{in}}$, it exists a value $\Delta k = k_{\text{in}} - k_{\text{out}} = \Delta^*$ such that, below this threshold, no algorithm can capture the community structure better than by chance:

$$\Delta^* = 2\sqrt{k}. \tag{8}$$

In the following, to measure the quality of the partition computed by our models, we use the adjusted Rand index (ARI) [37] defined by:

$$\text{ARI}(g_t, g_e) = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{N}{2}}{\frac{1}{2}\left[\sum_i\binom{a_i}{2}+\sum_j\binom{b_j}{2}\right] - \left[\binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{N}{2}}, \tag{9}$$

where, index i goes over all the sets in g_t, while index j goes over all sets in g_e.

n_{ij} defines the number of elements in common between the sets in $g_t(i)$ and $g_e(j)$, $a_i = \sum_j n_{ij}$ is the sum of the elements in common between set i and all sets in j, and $b_j = \sum_i n_{ij}$ is the sum of the elements in common between the set j and all the possible sets j. Hence, the value of ARI varies between 0, when none of the elements is well classified, and 1, when there is a perfect matching between the real membership and the one inferred.

As a first experiment, we use the modularity function defined with the null model in Eq. (6), denoted as:

$$Q(g) = \sum_{i,j}\left[a_{ij} - \frac{x_i x_j}{1+x_i x_j}\right]\delta_{g(i)g(j)}. \tag{10}$$

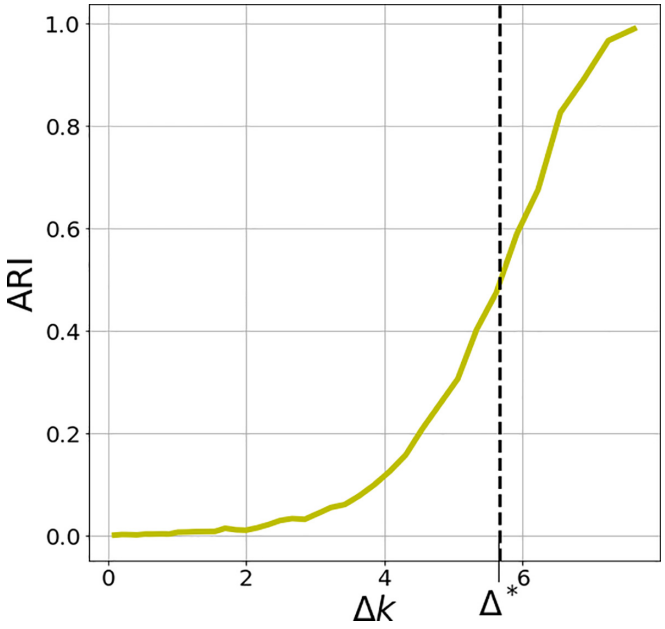

FIG. 1. Figure shows the performances using the modularity defined in Eq. (10) on a static network with $k = 8$ and 200 nodes. Each point is averaged over 30 iterations.

To optimize the partitions, we use a modified version of the algorithm introduced by Clauset and Moore, as detailed in Appendix 4. We explore how the modularity performs by varying the relationship between p_{in} and p_{out}. Interestingly, since the expected degree is the same for each node, a suitable null model for our case involves a single parameter, in contrast to the CM that requires N parameters. However, the choice of the null model does not significantly affect the final results, so we employ the binary configuration model for illustrative purposes.

Figure 1 displays the transition between a regime where the modularity function-based algorithm successfully detects the community structure and a regime where it fails. The plot results from examining of graphs, each composed of two communities, where the relationship between p_{in} and p_{out} varies, implying a variation in Δk as previously described. These artificially generated graphs maintain a constant average node degree k. The plot substantiates the theoretical forecast that a distinct transition point, denoted as $\Delta k = \Delta^*$, exists. As can be observed, our algorithm, consistent with theoretical expectations, performs better than expected, as measured by the ARI up to the detectability threshold. It is important to note that, since the networks we are estimating are of finite size, we cannot completely eliminate random fluctuations, which result in peculiar algorithmic performance behavior near the phase transition. This effect has been noted in several studies [34,38,39]. Indeed, near this threshold, finite size effects can make the phase transition not abrupt, but rather smooth and continuous. Thus, we will assume the numerical simulation is consistent with the theoretical result when the ARI value is close to 0.5. As we will see in the rest of the paper, this value agrees with other well-documented results in the literature, supporting our assumption. Throughout this paper, all numerical simulations designed to demonstrate changes in the DT

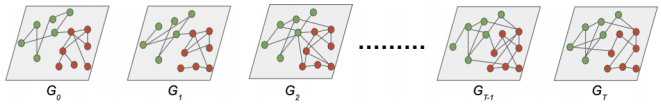

FIG. 2. Schematic representation of temporal networks with fixed communities. Each node in the network is assigned a color to indicate its membership, which remains fixed throughout the evolution of the network.

depending on the model used will have the same parameter values; thus, they will also be characterized by these effects near the DT. Finite-size effects can be mitigated by increasing the number of elements in the simulation. The main problem is the complexity involved in solving parameter estimation for the more complex null models addressed in this work. As the number of nodes increases, the computation time increases exponentially, making the problem infeasible. This is why we have opted to work with smaller networks.

III. MODULARITY FOR TEMPORAL NETWORKS WITH A STATIONARY DATA GENERATING PROCESS

The transition from static to temporal graphs introduces complexities that significantly affect community detection tasks. As the network evolves, changes in community structures may occur [40]. To address these challenges, selecting a representation method that aligns with our algorithm for analyzing temporal networks is crucial. In this study, we use a snapshot approach, representing the network as a sequence of T static sparse graphs, denoted as G_i. Each graph in this trajectory, referred to as a graph trajectory, $\mathcal{G}$, maintains a constant number of nodes, N, and is characterized as unweighted and undirected:

$$\mathcal{G} \equiv \{\mathbf{G}_1, \ldots, \mathbf{G}_T\}. \tag{11}$$

As previously mentioned, communities in temporal networks undergo a variety of transformations [41]. In the theoretical part of this work, we focus on persistent communities that remain constant over time, see Fig. 2. Although this scenario is relatively simple, it is worth investigating because, as we show in the following sections, it allows us to develop insights into how memory impacts community discovery. Afterward, we explain how to adapt our approach to situations where communities evolve throughout the network's lifespan. In the following sections, we examine graph trajectories generated with communities that maintain consistency over time, produced through a stationary process. Specifically, we explore two different data-generating processes (DGP): initially, a time-varying graph with no correlation between snapshots, which then evolves into a temporal network with inherent memory.

A. Temporal networks with independent observations

The first DGP we analyze produces a graph trajectory consisting of independent and sparse snapshots. Each snapshot is constructed using the planted partition model, which is the same model applied in the static case. In this configuration, the average connectivity for each block within each snapshot remains consistent at k, ensuring that the average connectivity

over time, $\sum_t \frac{1}{T}k(t)$, equals $\bar{k} = k$. Consequently, the model defined has the following expression for the probability of observing a given graph trajectory $\mathcal{G}$:

$$P(\mathcal{G}|g) = \prod_{t=1}^{T}\prod_{(ij)} p_{g_i g_j}^{a_{g_i g_j}(t)}(1 - p_{g_i g_j})^{1-a_{g_i g_j}(t)}. \tag{12}$$

We refer to this model as a dynamic stochastic block model (DSBM).

To address community detection within this framework, we employ a modularity function that utilizes a naive temporal generalization of the binary configuration model. This model is constructed by replacing the degree sequence of a single snapshot with the temporal average as a constraint. Consequently, the related Hamiltonian is expressed as follows:

$$H(\mathcal{G}|\vec{\alpha}) \equiv \sum_{i=1}^{N}\alpha_i \sum_{t=1}^{T}\frac{k_i(t)}{T}. \tag{13}$$

Formally, the resulting expression for the expected value of the link existence between two nodes is the same as the non-temporal CM [20], but this time it refers to its average over time:

$$\langle\overline{a_{ij}}\rangle = \frac{x_i x_j}{1 + x_i x_j}, \tag{14}$$

Where, in this case, $x_i = e^{-\alpha_i/T}$.

Therefore, the related modularity can be defined as:

$$Q(g) = \sum_{i,j}[\,\overline{a_{ij}} - \langle\overline{a_{ij}}\rangle\,]\delta_{g(i)g(j)}, \tag{15}$$

with $\langle\overline{a_{ij}}\rangle$ as in Eq. (14) and $\overline{a_{ij}} = \frac{1}{T}\sum_T a_{ij}(t)$. Intuitively, when the community structure does not change in time, as the case described in Eq. (12), using Eq. (15) allows for the filtering out spurious links by observing several realizations of the same network, therefore directly putting more evidence on the hidden block structure.

However, by employing this modularity function, we implicitly assume that each snapshot represents an independent realization of the same network. This approach overlooks the temporal order in the graph's trajectory, thereby discarding relevant information when it is present.

B. Community detection in Markovian time-varying graphs

We now aim to introduce a refined version of the modularity function that incorporates node features related to their capacity to maintain links. These ideas originate from the findings in Ref. [20], where the authors highlight the significance of node-level memory and its critical role in analyzing real-world systems. With this in mind, we demonstrate how leveraging these features can enhance the performance of the modularity function in community detection tasks. To achieve our objectives, we employ a DGP with a Markovian structure, allowing for controlled memory incorporation into the system.

The DGP we employ is conceptualized as a dynamic stochastic block model, distinct from similar models in the literature, such as those described in Refs. [17,22,42]. Unlike other DGPs where memory is an external parameter independent of the network's intrinsic characteristics like degree distribution, our model integrates a memory effect directly. This effect is realized through a Markovian evolution of the system, where node-specific features govern interdependently both the formation and persistence of links. To introduce this model, we first identify the features to be encoded in the maximum entropy model, which forms the basis for both the DGP and the definition of modularity. In this context, we focus on the persisting degree, which represents the portion of the degree maintained between one snapshot and the next:

$$h_i(t,1) \equiv \sum_{j\neq i} a_{ij}(t)a_{ij}(t+1) = \sum_{j\neq i} b_{ij}(t,1). \tag{16}$$

We will refer to the quantity $a_{ij}(t)a_{ij}(t+1)$ as the one-lagged persisting link.

Hence, imposing the persisting degree sequence as further constraints in the case described by the Hamiltonian in Eq. (13), we generate the following Hamiltonian:

$$H(\mathcal{G}|\vec{\alpha},\vec{\beta}) \equiv \sum_{i=1}^{N}\left[\alpha_i\sum_{t=1}^{T}\frac{k_i(t)}{T} + \beta_i\sum_{t=1}^{T}\frac{h_i(t)}{T}\right]. \tag{17}$$

By solving the related problem [20], we can obtain the analytical expression for the expected value of the average link and the average persisting link:

$$p_{ij} = \langle\overline{a_{ij}(t)}\rangle, \tag{18}$$

and

$$q_{ij} = \langle\overline{a_{ij}(t)a_{ij}(t+1)}\rangle. \tag{19}$$

Both the expressions as function of $\vec{\alpha}$ and $\vec{\beta}$; see Appendix 1 for details.

Here, p_{ij} represents the marginal probability of having a link between node i and node j at time t, and

$$P[a_{ij}(t+1) = 1 \;\&\; a_{ij}(t) = 1] = q_{ij}, \tag{20}$$

is the probability that two nodes have a connection at time t and $t+1$. Once obtained p_{ij} and q_{ij}, we can compute the following stochastic matrix:

$$P_{ij}(q_{ij}, p_{ij}) = \begin{bmatrix} \frac{q_{ij}}{p_{ij}} & \frac{p_{ij}-q_{ij}}{p_{ij}} \\ \frac{p_{ij}-q_{ij}}{1-p_{ij}} & \frac{1-2p_{ij}+q_{ij}}{1-p_{ij}} \end{bmatrix}, \tag{21}$$

which defines the transition matrix associated with the evolution of each link. This stochastic matrix essentially defines the system's evolution, ensuring that the expected values for the persisting degree and the degree precisely match those constrained in Eq. (17). This matrix is applicable only for a link that exists in at least one snapshot of the graph trajectory, i.e., $p_{ij} \neq 0$.

Operationally, our implementation of the dynamic stochastic block model starts by generating an initial network snapshot using a static planted partition model. Afterward, the presence of links evolves according to the stochastic matrix detailed in Eq. (21), creating the full trajectory.

Similar to the other cases, in our experiments, we generate graph trajectories by specifying our DGP with two distinct probabilities: p_{out} and p_{in} for connection, and q_{out} and q_{in} for preserving links. These probabilities automatically induce an average degree, k, and an average persisting degree, h, respectively. Consequently, the membership influences both

the link density within blocks and the persisting link density. All probabilities are set to ensure the network's sparsity is maintained, being of the order $O(\frac{1}{N})$, which is a crucial condition for defining the expression for the DT.

Once the graph sequence $\mathcal{G}$ is generated, we can extract two sequences of matrices:

$$\mathcal{A} = [A(1), A(2) \dots A(T)], \tag{22}$$

and

$$\mathcal{B} = [B(1), B(2) \dots B(T)], \tag{23}$$

where $A(t) = [a_{ij}(t)]$ and $B(t) = [a_{ij}(t+1)a_{ij}(t)]$, from which we can define a new matrix

$$C(t) = A(t) + B(t), \tag{24}$$

such that each element of $c_{ij}(t) = a_{ij}(t) + b_{ij}(t)$.

By computing the average in time of $A(t)$ and $B(t)$, we obtain the matrix $\overline{C}$, whose elements are used to define a new modularity in the following way:

$$Q(g) = \sum_{ij \in N} [\overline{c_{ij}(t)} - \langle \overline{c_{ij}} \rangle] \delta(g_i, g_j), \tag{25}$$

where $\langle \overline{c_{ij}} \rangle = p_{ij} + q_{ij}$. Through this approach, we account for two contributions that are, in principle, interdependent and capable of providing distinct insights into the DGP.

IV. DETECTABILITY THRESHOLD FOR STATIONARY TIME-VARYING GRAPHS

This section illustrates how the DT varies based on the assumptions about the DGP and the type of modularity used. As previously stated, the DGP is constant over time, indicating that our data are samples from a stationary process. Under this premise, it is intuitive that an increase in the number of samples, i.e., the length of the trajectory, reduces the influence of noise on the observed block structures. This reduction occurs as noise is averaged out over more samples, consequently enhancing the DT and leading to improved outcomes. Moreover, as we observe in the numerical simulations, the finite-size effect, already shown in Fig. 1, tends to be filtered out as well with an increasing number of snapshots, characterizing a sharper transition near the DT.

Nevertheless, this intuition only tells part of the story. As we will demonstrate, when the persistence of connections is a relevant topological feature for characterizing community membership, using a modularity function that incorporates this information can positively alter the expression for the DT. Consequently, this enhances the configuration space in which the presence of communities can be detected. Conversely, using a function that does not consider this feature may result in worse detection.

A. Detectability threshold for noncorrelated time-varying graphs

We now revisit the expression for the DT in a well-known scenario, where we consider a DGP in which each snapshot is sparse and independent, as outlined in Eq. (12). It is established that under these conditions, an expression for the DT has been derived by Ref. [22] for $T \to \infty$ and has been conjectured for finite T, a conjecture that was later analytically supported in Ref. [42]. The computation focuses on the sparsity of each snapshot, which suggests a locally treelike structure for the graph trajectory $\mathcal{G}$. Consequently, the authors in Refs. [22,42] have interpreted the community detection task in temporal networks as a label recovery problem. They demonstrate that, in our specific case where communities remain stable over time, community detection is feasible only if:

$$\Delta_\nu^* = 2\sqrt{\frac{\overline{k}}{T}}, \tag{26}$$

where ν indicates no-memory in the DGP.

The type of temporal network defined in Eq. (12) is designed to incorporate topological information of membership based solely on the density of connections, without considering other features such as their persistence. Intuitively, this implies that the DT has an expression that is a function of this quantity, described towards the degree. Interestingly, as anticipated, the threshold is also a function of T and is such that as T increases Δ_ν^* goes to zero, so, if T is big enough, we will always spot the block structure. Given this framework, here we can employ a modularity function that uses only information on the degree sequence to spot the community structure.

B. Detectability threshold for Markovian time-varying graphs

Transitioning to Markovian temporal networks introduces a series of issues that negatively affect various aspects of our approach so far. Introducing snapshot correlations complicates the observed matrix sequence $\mathcal{A}$, with the presence of links influenced not only by the community structure but also by memory effects, which have been shown to affect community detection performance adversely [17]. Furthermore, the lack of independence between consecutive snapshots undermines the prerequisites for computing the DT, as described in [22,42], preventing an explicit phase transition prediction.

Indeed, the aspects highlighted above are interconnected and can be well-described within our framework. Specifically, using the stochastic matrix in Eq. (21), that defines a fundamental part of our DGP, we can precisely determine the necessary waiting period to consider two consecutive matrices as uncorrelated. This adjustment allows us to restore the conditions required to calculate the threshold and to quantify how the algorithm's performance deteriorates in comparison to the related non-Markovian case.

Hence, by acknowledging that each link present in at least one snapshot evolves according to the stochastic matrix defined in Eq. (21), we can extract two eigenvalues from each matrix. The first eigenvalue, by definition, is equal to 1, and the second is given by:

$$\lambda_{ij}^{(2)} = \frac{q_{ij} - p_{ij}^2}{p_{ij} - p_{ij}^2}. \tag{27}$$

This eigenvalue is directly linked to the system's memory. Indeed, by calculating $\lambda_{ij}^{(2)}$, as detailed in Ref. [20], we can determine the correlation length and the rate at which the correlation diminishes between two consecutive snapshots.

Consequently, we obtain τ_{ij}^*, which is defined as:

$$\tau_{ij}^* = \frac{1}{\ln\left(\frac{1}{\lambda_{ij}^{(2)}}\right)}. \tag{28}$$

Hence, waiting for a duration equal to the maximum of all τ_{ij}^*, denoted as τ^*, between one snapshot and the next ensures that the snapshots can be considered independent. At this point, by discounting the entries of the matrix $\overline{\mathcal{A}}$ by the maximum correlation length τ^*, we reproduce the condition for the calculation of the DT for finite T, resulting in the following expression for the DT:[1]

$$\Delta_{M,v}^* = 2\sqrt{\frac{\tau^*}{T}\bar{k}}, \tag{29}$$

where M stands for Markovian, and indicate that the threshold refers to a Markovian DGP.

Moving ahead, we now adopt a perspective in which information about the existence of communities is not only confined to the density of connections but also extends to how these connections persist over time. In scenarios where the process exhibits memory, relying solely on the degree sequence to identify communities results in the neglect of crucial information that could provide deeper insights into the block structure. In the following sections, we demonstrate how incorporating the density of persistent connections, through the persisting degree, into the expression of modularity, enhances the algorithm's performance and alters the expression for the DT.

In this regard, our innovative modularity function, as defined in Eq. (25), integrates information from two distinct matrices, $\overline{\mathcal{B}}$ and $\overline{\mathcal{A}}$. This approach allows us to present how the method proposed by Ghasemian *et al.* [22] to compute the DT can be re-adapted to our context, thereby implying a new and better threshold.

We have previously demonstrated that, by adjusting $\overline{\mathcal{A}}$ with τ^*, its elements can be interpreted as those realized by an equivalent DSBM with independent snapshots [17,42]. To extend this rationale to the entire matrix $\overline{\mathcal{C}}$, it is necessary to justify how the same discounting factor can be applied to the matrix $\overline{\mathcal{B}}$. This is achieved by explicitly calculating the autocorrelation between elements at time t and $t+1$ for sequence $\mathcal{B}$. As detailed in Appendix 2, by adjusting for a time τ^*, even in this scenario, the elements of sequence $\mathcal{B}$ can be considered independent. This adjustment allows for the application of the previously discussed methodology in calculating the DT. Thus, when considering the matrix resulting from the sum of $\overline{\mathcal{A}}$ and $\overline{\mathcal{B}}$, we re-establish the conditions necessary for the analytical computation of the DT:

$$\Delta_{M,\mu}^* = 2\sqrt{\frac{\tau^*}{T}\bar{c}}, \tag{30}$$

where $c = \bar{k} + \bar{h}$, and $\bar{h} < \bar{k}$ by definition. The symbol μ as a subscript indicates memory, emphasizing that the modularity utilized for this data set, which exhibits Markovian dependency, leverages memory. Although the threshold indicated in Eq. (30) is always larger than the one indicated in Eq. (29), it should be noted that, while in the case of Eq. (29), the space where the communities are observable is essentially delineated by values of Δk greater than the threshold, in the case of Eq. (30), in addition to the values of Δk, the values of $\Delta h = h_{\text{in}} - h_{\text{out}}$ also contribute. This makes the choice of this second approach always preferable, as we will see in the next section.

[1]All the links for which this equation is valid exist in at least one snapshot and have $p_{ij} \neq 0$.

V. NUMERICAL SIMULATIONS

To validate the analytical results presented in the previous sections, we have conducted a series of numerical simulations. These simulations demonstrate the consistency between the predicted and observed thresholds. Additionally, we characterize the algorithm's performance across various combinations of parameters, showing how our modularity function works better in different scenarios.

To achieve this, we utilize the DGPs previously introduced for temporal networks, along with their associated modularity functions. For all experiments, we optimize the modularity function using the algorithm detailed in Ref. [43], adapting the expression for the null model specific to each case, see Appendix 4. We generate the trajectories by varying the values of Δk and Δh, as defined in:

$$\Delta k = k_{\text{in}} - k_{\text{out}} \quad \text{and} \quad \Delta h = h_{\text{in}} - h_{\text{out}}, \tag{31}$$

with

$$k_{\text{in}} = \frac{N}{2}p_{\text{in}}, \quad h_{\text{in}} = \frac{N}{2}q_{\text{in}}, \tag{32}$$

$$k_{\text{out}} = \frac{N}{2}p_{\text{out}}, \quad h_{\text{out}} = \frac{N}{2}q_{\text{out}}. \tag{33}$$

Here, N denotes the number of nodes within a single snapshot. In general, each trajectory is generated to maintain both the average degree and the average persisting degree across the entire temporal network, when those are specified, or otherwise only the average degree. In this setting, Δk signifies the disparity between the average (temporal) number of links formed within the communities and those outside, whereas Δh denotes the difference in the persistence of links within these communities compared to external links. In all the experiments, we have two communities whose elements are known and play the role of ground truth.

The initial series of experiments involves generating graph trajectories with independent observations, as in Eq. (12), by varying k_{in} and k_{out}, thus altering the value of Δk while maintaining a constant average degree $\bar{k}$. By specifying the number of time steps for each trajectory, we consequently establish the threshold Δ_v^*. We generate 50 trajectories for each Δk value and various T values, then plot the average performance across these 50 instances.

In Fig. 3, the results obtained using the modularity function described in Eq. (15) are represented. The red vertical line corresponds to the static DT, as computed in Eq. (8) based on the first snapshot of the trajectory (see Fig. 1). Interestingly, even with a few snapshots, the improvement in performance quality is evident. In all cases, the DT expected from theory and those computed numerically are the same. Even in these cases, we observe the presence of finite-size effects, as previously noted

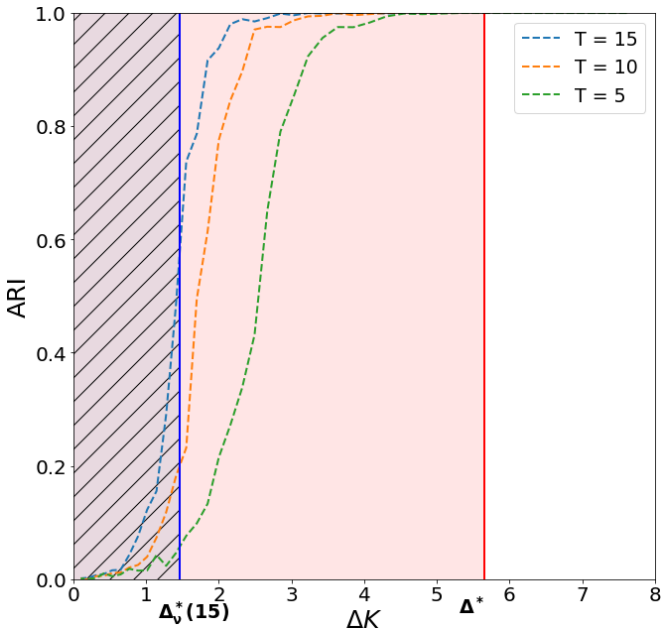

FIG. 3. This figure illustrates three different performance profiles of the modularity, as described in Eq. (15), for graph trajectories of varying lengths generated using the DSBM outlined in Eq. (12), for different valued of Δk. The vertical blue line represents the DT for the temporal network with $T = 15$, while the red vertical line indicates the DT for a single static snapshot. For each trajectory, we maintain $\overline{k} = 8$, two communities and $N = 200$. Each point is the result of the average over 50 sampled trajectories.

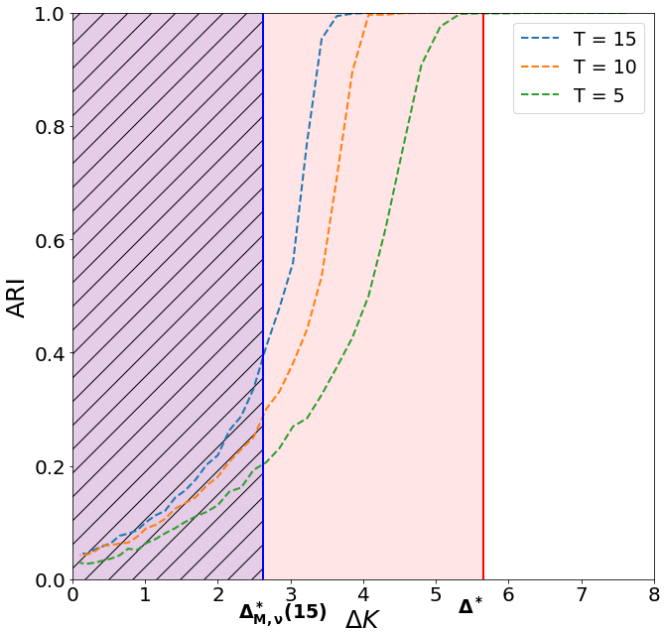

FIG. 4. This figure illustrates three different performance profiles of the modularity, as described in Eq. (15), for graph trajectories of varying lengths generated using the DSBM outlined in Sec. III B, for different valued of Δk. The vertical blue line represents the DT for the temporal network with $T = 15$, while the red vertical line indicates the DT for a single static snapshot. For each trajectory, we maintain $\overline{k} = 8$, $\overline{h} = 4$, two communities and $N = 200$. Each point is the result of the average over 50 trajectories.

in Sec. II B. It is also noteworthy that as the number of snapshots, T, increases, the finite-size effects become less evident, supporting the assumption that as the number of iterations increases, these effects are gradually averaged out.

Our next analysis explores graph trajectories generated through a Markovian data-generating process, as detailed in Eq. (21). Similar to our previous experiments, we generate 50 trajectories for different values of T and Δk, and then we average the performances for each trajectory. However, in this updated scenario, we incorporate a memory effect by keeping the values of q_{in} and q_{out} constant but allowing p_{in} and p_{out} to vary. This approach, as discussed in Sec. IV B, is expected to significantly influence the performance of the modularity, previously defined in Eq. (15).

Figure 4 illustrates the adverse effects of memory on the quality of the detected communities. This diminishes the benefits of using additional snapshots compared to the static case, thereby confirming the observations in Eq. (29). Notably, the curve patterns in Fig. 4 differ from those observed in Fig. 3. Intrinsically, the value of τ^*, which reflects the memory level in our process, given the way the trajectory is built, is influenced not only by q_{in} and q_{out}, that are kept fixed in this experiments, but also by Δk, as expressed in Eq. (27). Hence, the variation of Δk (p_{in} and p_{out}), impacts memory and alters the shape of curves.

Although this is the case, we can easily predict when the value of Δk of the trajectory surpasses the expected DT, and so indicate the point at which we are expecting the presence of the change in the algorithm performances. As shown in Fig. 4, our prediction captures those points well. It is worth noting that the finite-size effect, together with the dynamic behavior of the memory just described, produces a much smoother transition than the one observed in Fig. 3, making the predicted point less evident than its memoryless counterpart.

We now take a further step by demonstrating how integrating information on community structure into link persistence can help recover some of the information about community structure that is lost due to the presence of memory. We proceed to the most comprehensive scenario addressed in this study, generating graph trajectories while varying Δk and Δh for two different values of T, $T = 15$ and $T = 50$. It is important to note that the average value of $\overline{h}$ should remain lower than $\overline{k}$ during trajectory generation to create realistic trajectories. Moreover, as we adjust Δk and Δh while maintaining the values of $\overline{k}$ and $\overline{h}$ constant, we are indirectly changing τ^*, which affects both the curve shapes and the value of the DT as previously mentioned and shown in Fig. 4.

To accomplish this task, we employ the dynamic stochastic block model described in Sec. III B. In this scenario, we compare the modularity that does not consider temporal dependency, Eq. (15), with our proposal, Eq. (25). We verify whether the expected DTs, $\Delta^*_{M,\mu}$ and $\Delta^*_{M,\nu}$, are accurately reproduced.

The process used to generate the trajectories ensures that the evolution does not change the relationship between p_{in} and p_{out} when altering Δh, facilitating a direct comparison between the modularity built on the model with memory and the one built on the memoryless model.

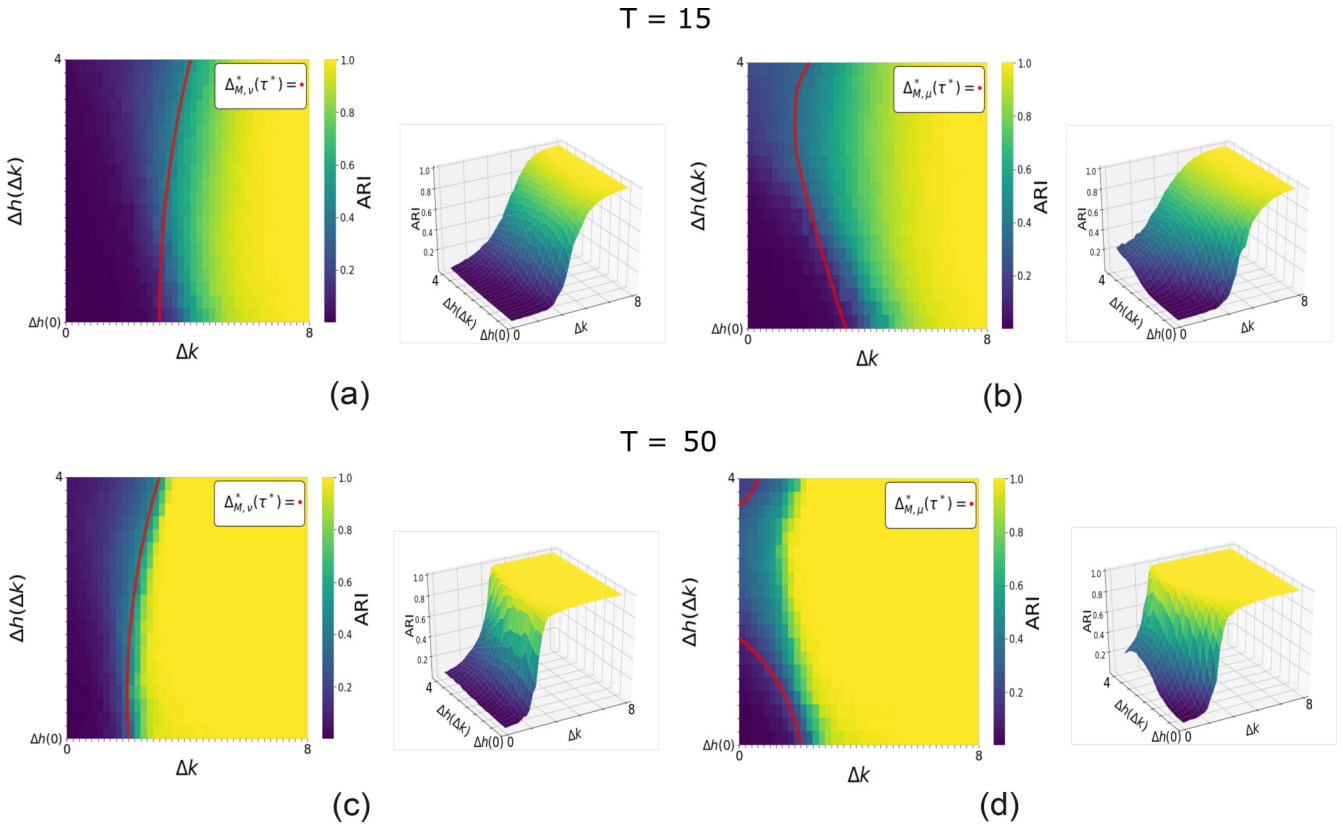

FIG. 5. This figure presents the results of two different types of simulations, each employing distinct modularity equations. The first uses the modularity equation from Eq. (15) and the second utilizes the modularity equation from Eq. (25). All trajectories are generated according to the model described in Sec. III B, and each point represents the average over 20 trajectories, simulated with corresponding values of Δk and Δh. (a) illustrates the results from applying the modularity equation (15) to graph trajectories with $T = 15$, while (b) relates to the application of the modularity equation from Eq. (25), on the same trajectories. (c) and (d) apply the same modularity equations as (a) and (b), respectively, but with $T = 50$. All generated trajectories have $N = 200$, $\bar{k} = 8$, $\bar{h} = 4$, and consist of two communities. Red lines represent the analytically derived DT.

The experimental outcomes, as depicted in Fig. 5, align with expectations. Figures 5(a) and 5(c) demonstrate the application of the modularity Eq. (15) for trajectories generated with 15 and 50 snapshots, respectively. As expected, more snapshots lead to a more precise inference of the community structure given the same set of parameters. This results in an expanded space of parameters where the algorithm can effectively capture the presence of communities, as previously observed in Fig. 4. The detectability threshold, $\Delta^*_{M,\nu}$, changes with the presence of memory, which varies with Δh. Increasing Δh not only impacts the value of τ^* but also the behavior of the curve near the DT, making it smoother, as observed in the three-dimensional (3D) plots. Figures 5(b) and 5(d) showcase the application of the novel modularity function (25) to the same trajectories used in Figs. 5(a) and 5(c). In these figures, the threshold exhibits peculiar behavior due to the combination of changes in memory resulting from variations in Δk and Δh. These changes imply alterations in the values of p_{in}, p_{out}, q_{in}, and q_{out}, upon which the detectability threshold ($\Delta^*_{M,\nu}$) depends through τ^*. An interesting phenomenon to note is the behavior of the threshold as Δh increases. As expected, as Δh increases, the algorithm more effectively distinguishes communities. However, this increase also elevates the value of τ^*, eventually offsetting the benefits gained by increasing Δh. Comparing the plots where the modularity function includes memory versus those that do not, our modularity (25) consistently outperforms the other in recognizing communities. The no-memory approach fails to recognize the varying propensity of nodes within different groups to preserve links, inadvertently discarding valuable information that is better captured by our approach. The only scenarios where the distinction between modularity incorporating the persisting degree and that without it becomes negligible are those where $\Delta h(0)$.

VI. PROXIMITY NETWORKS

We now examine a real-world application, aiming to test the implementation of the previously defined modularities on a data set where ground-truth community information is accessible. This data set provides information about proximity relations between students and teachers in a primary school [23]. Electronic proximity detectors worn by the participants were used to collect the data, capturing the presence of other nearby detectors at 20-s intervals. The data was collected over a period of two days, specifically from October 1 to October 2, 2009. For simplicity, in this analysis, we focus only on the data from the first day. Based on this data, we aim to

construct a temporal network, which will be characterized by a series of snapshots created by aggregating data over a specific time window. This implies that within each window, two nodes will be connected if they have come into contact at least once during that time interval. In our study, we leverage verified information about existing communities. The underlying hypothesis is that the network dynamics are primarily driven by the community memberships of the nodes. The aim is to investigate whether, by examining these dynamics, it is possible to discern the node memberships. It is important to note that these dynamics are far from trivial, as even during class hours, students change classrooms and have the freedom to move around.

Working with snapshot representations of real temporal networks requires taking into account how the choice of time windows for aggregating information can impact the quality of results, as noted by Krings *et al.* [44] and Clauset *et al.* [45]. Before suggesting a solution to this challenge for our specific scenario, we begin analyzing the real network by initially adopting a 10-min time aggregation window. This initial aggregation will be used for preliminary analyses, which will lay the foundation for developing a strategy to determine the optimal window for our task.

The proposed approach for community identification is based on the assumption that the process that generated the data is stationary. Therefore, it becomes crucial to identify temporal segments for which it makes sense to consider the network as generated by the same process. To solve this task and identify potential breakpoints in the system's evolution, we apply the structural breaks approach introduced in Ref. [20]. This approach uses a model that considers memory at the node level as the reference model; see Appendix 5 for further details. It is noteworthy to highlight that using the structural breaks algorithm enhances the applicability of these types of modularity functions in real-world systems, allowing us to deal with networks that exhibit evolution in the community structure.

To evaluate the ability of our algorithms to capture communities in time, we assume that the real-world network is generated using a dynamic stochastic block model using the ground truth communities, and with respect to this, we measure the performances. This involves observing the algorithm's behavior and assessing the quality of the communities identified in various segments.

Figure 6 presents the performance of our algorithm executed over a 10-min time window, identifying several breakpoints that reflect diverse behaviors. In this scenario, the dynamics are attributed to students' class memberships and their movements between classrooms. Notably, our approach indicates that the influence of class-based membership significantly diminishes during playground periods. This observation suggests that the supposed DGP, a dynamic stochastic block model used to form the trajectory, accurately describes behavior only during class hours. Conversely, the decreased quality of the communities observed during playground time compared to the ground truth suggests that the assumed DGP may not be correct for this interval, indicating that this model does not adequately capture the dynamics.

In the last part of our real-world application, we investigate how the quality of the communities identified changes by

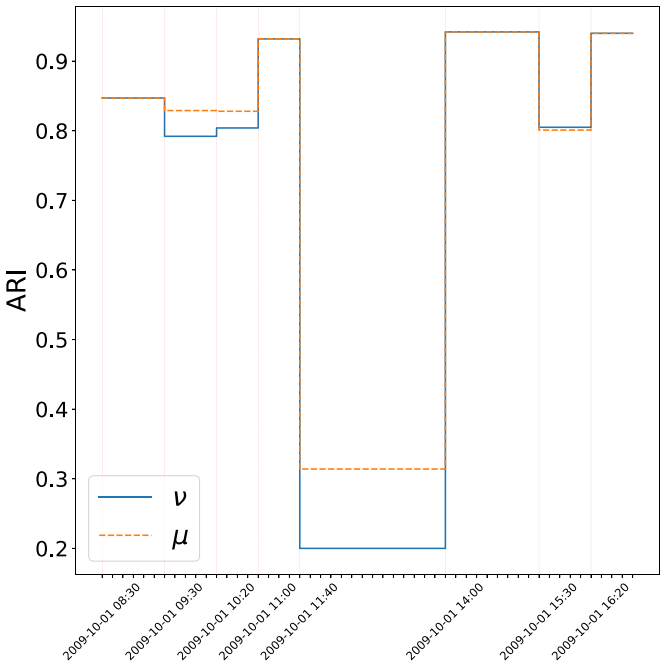

FIG. 6. The figure displays the ARI values across 10-min temporal windows at various identified breakpoints (red vertical lines). Each illustrated segment is proportional in length to the period during which the network can be considered stationary. Within this interval, the ARI value is fixed and calculated by comparing the detected communities to those presumed real. The figure showcases the results for two different types of modularity functions.

choosing different time windows to define the temporal network. Our objective is to determine if there is a natural scale where the effects of membership on link density and persistent link density are more evident. To this end, we distinguish between class hours, where a higher similarity between the discovered communities and the ground truth was noted, and playground hours. Our strategy is applied to the trajectories built upon various time windows. For each time window, we identify distinct stationary segments. In each segment, we assess the ARI between the communities detected using our modularities and the ground truth. Additionally, we characterize each stationary segment using the measured values of Δk and Δh.

In this context, we define Δk as the difference between the average (temporal) number of links formed within the ground-truth communities and those formed outside them. Similarly, Δh represents the difference in the number of links that persist within the ground-truth communities compared to those outside. Consequently, we can write the terms in Eq. (31) become:

$$k_{\text{in}} = \sum_{ij} \overline{a}^*_{ij} \delta(g^*_i, g^*_j), \quad h_{\text{in}} = \sum_{ij} \overline{b}^*_{ij} \delta(g^*_i, g^*_j), \tag{34}$$

$$k_{\text{out}} = \sum_{ij} \overline{a}^*_{ij} I_{(g^*_i \neq g^*_j)}, \quad h_{\text{out}} = \sum_{ij} \overline{b}^*_{ij} I_{(g^*_i \neq g^*_j)}, \tag{35}$$

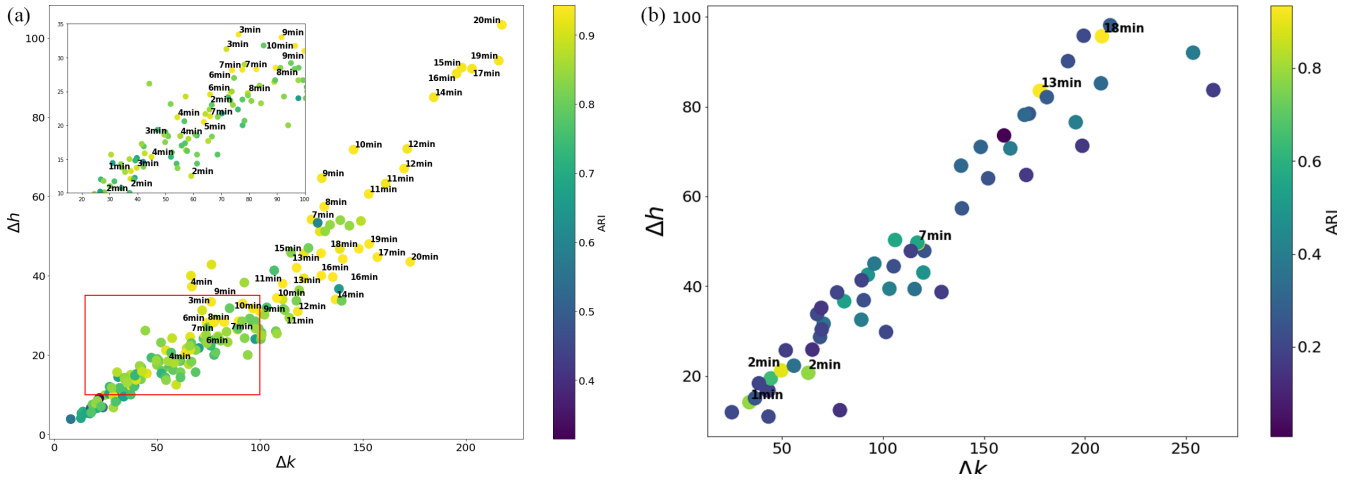

FIG. 7. Figure displays the performance of community detection on a real network, where each point corresponds to a stationary segment identified using structural break detection, see Appendix 5. Each point is assigned values of Δk and Δh, and the color of the points indicates the ARI value, which is calculated by comparing the detected communities, using the modularity with memory, against the ground truths within the specific segment. In (a), the values during class hours are shown, while (b) depicts values during playground time. The label associated with each point indicates the size of the temporal window used to create the temporal network for that segment. Only labels corresponding to ARI values greater than 0.8 are displayed.

where the $*$ indicates the real values, and with I as indicator function

$$I_{(g_i \neq g_j)} = \begin{cases} 1 & \text{if } g_i \neq g_j, \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

In Fig. 7, it is shown that during class hours, an increase in both Δk and Δh corresponds with a rise in the ARI. However, this correlation does not appear during playground hours, as previously observed in Fig. 6. Unlike scenarios involving only two partitions, Δk and Δh in this context do not provide exactly the same insights. We observe the cumulative sum computed across all possible clusters, each containing more than two entities in our dataset. Therefore, the increase in the difference between links may be attributed to the influence of one specific community rather than being uniformly distributed across all clusters.

Our results indicate that for the nonplayground hours, performances, regardless of the aggregation window size, align with expectations for a scenario where evolution is genuinely driven by membership. As the temporal window increases, there is a corresponding increase in Δk and Δh for certain segments, which the model can leverage to enhance the quality of community detection. During these hours, there is a noticeable correlation between memory and community structures.

Conversely, the situation during playground hours is different. In these instances, additional factors likely intermingle with the existing dynamics, confirming the presumed dynamic stochastic block model being less effective in describing the system.

We propose further analysis to explore the relationship between memory and the quality of detected communities. Our hypothesis posits that if the membership of its elements drives the system's evolution, a natural temporal scale might exist where both the quality of communities, measured by the ARI, and memory reach their peak. Identifying this scale could determine the optimal time window for data aggregation, making the communities easier to distinguish.

To measure memory, we use normalized autocovariance computed on the links persistence. This normalization controls for the effects of degree since higher degrees generally lead to higher average autocovariance. By dividing by the degree of each node, we mitigate this effect.

Therefore, as a metric, we employ the average autocovariance, defined as follows:

$$\overline{\mathbf{A}}(W, \tau = 1) = \frac{1}{N} \sum_{i}^{N} \frac{A_i(\tau = 1)}{k_i}. \tag{37}$$

Here, the average is computed over all nodes N in the network, and

$$A_i(\tau) = \frac{1}{T} \sum_{t=1}^{T} [h_i(t, \tau) - \langle h_i(t, \tau) \rangle_v], \tag{38}$$

represents the autocovariance specific to node i. In this context, $\langle h_i(t, \tau) \rangle_v$ denotes the expected node degree persistence according to a null model that precludes memory. To accomplish this, we utilize a null model designed not to contemplate memory effects, as defined in Eq. (14). This approach retains genuine temporal correlations between snapshots.

With class hours identified as the segments where dynamics are driven by membership, our focus shifts to these periods for examining the behavior of normalized autocovariance. We vary the time window within a range from 1–20 min, allowing us to explore how the autocovariance responds to different temporal aggregations:

$$W \in \{1', 2', 3', \ldots, 20'\}. \tag{39}$$

We identify all the stationary segments that fall within class hours using the same approach as before, corresponding to the different time windows. For each of these, we calculate the normalized autocovariance for $\tau = 1$. We use the average of the normalized autocovariance across all identified stationary segments as a measure of memory. Thus, defining *brks* as the set of segments where we can assume the memory to be

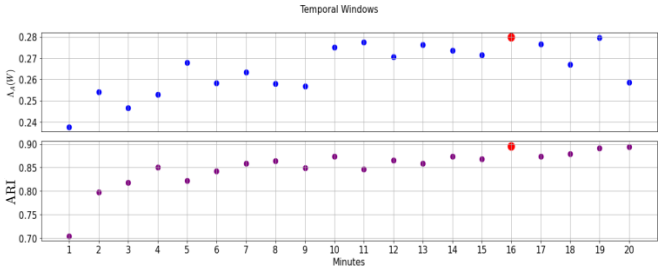

FIG. 8. The top plot displays the values of Λ_A for different time windows, while the bottom plot shows the values of the average ARI for the same time windows. The red circles indicate the maximum values for each series.

constant, the memory associated with each time window is defined as:

$$\Lambda_A(W) = \frac{1}{|brks|} \sum_{s \in brks} \overline{\mathbf{A}_s}(W, \tau = 1). \tag{40}$$

Hence, once for each segment, is computed the expected value of the persisting degree relative to a memoryless model ($\langle h_i(t, \tau) \rangle_v$), we can compute Λ_A. We define the best window W_{best} as the value of W for which $\Lambda_A(W)$ is maximum:

$$W_{\text{best}} = \arg\max \Lambda_A(W). \tag{41}$$

in Fig. 8 is presented the measurement of the normalized autocovariance as a function of the time window and the average ARI computed over the segments in-class hours. Each point on the plot is designed for a nonoverlapping time window. The red points represent the maximum value taken between all the windows. Between these two quantities, we measure the Pearson correlation, funding a strong and significative correlation with value 0.73, demonstrating a relationship between the presence of memory and the quality of the communities found. In our data, $W_{\text{best}} = 16$ min, which also corresponds to the maximum on the ARI. In support of this assessment, we have included controlled experiments in the Appendix 6 where the system evolves precisely in accordance with the hypothesis tested above.

VII. DISCUSSION

Effective community detection in temporal networks necessitates the development of new, appropriate null models [46]. Such models provide a basis for comparison and enable the identification of dynamic structures in networks, thereby enhancing the quality of the discovered communities. In this work, we contribute to this area by demonstrating how using the right null model can lead to discovering hidden structures with higher precision.

We highlight the potential of modularity-based methods in community detection by utilizing maximum entropy null models to tailor their formulation to the specific challenges encountered. We propose a modularity function that acknowledges the persistence of connections among nodes within the same community. By focusing on the issue of the detectability threshold, our findings reveal that selecting an appropriate null model can substantially improve the quality of the identified communities. This underscores the critical role of accounting for memory effects when present and elucidates how incorporating memory alters the detectability threshold for community identification.

In real-world applications, we demonstrate the importance of considering memory in community discovery and how this feature can assist in indirectly related challenges, such as determining the ideal time window for data aggregation to develop a time-varying graph. We propose a data-driven methodology to identify the most suitable time window that reveals a more distinct delineation of community structures. Furthermore, we demonstrate that by integrating this approach with a change-point detection algorithm, we can track the evolution of communities over time, identifying when they are the primary drivers of system evolution and when they are not.

The results indicate that if memory is present but not adequately incorporated into the model, it can deteriorate the quality of the identified communities. However, if properly integrated into the algorithm used for community detection, it can become a fundamental advantage.

In numerous real-world systems, the influence of community membership on memory is a critical factor that cannot be ignored. Under these circumstances, a function that integrates memory effects, such as the one we propose in this study, is essential. It holds the potential to uncover new patterns that were previously hidden. By acknowledging and incorporating memory, our approach opens the door to revealing deeper insights into the dynamics of these systems.

ACKNOWLEDGMENTS

This work is supported by the European Union - NextGenerationEU - National Recovery and Resilience Plan (NRRP) - Mission 4 Component 2 Investment 3.1, through the project SoBigData.it: Strengthening the Italian RI for Social Mining and Big Data Analytics, IR000013, CUP B53C22001760006 and by the European Union - NextGenerationEU - National Recovery and Resilience Plan (NRRP) - Mission 4 Component 2 Investment 1.3 through the project RECON-NET: Reconstruction, Resilience and Recovery of Socio-Economic Networks CUP D63C22001310006 financed through the "Bando A Cascata" of the University of Pisa under the FAIR PE0000013 project; by D.G. acknowledges support from the Dutch Econophysics Foundation (Stichting Econophysics, Leiden, the Netherlands) and from the project NAEFR: Network Analysis of Economic and Financial Resilience (PRO3 Scuole), CUP D67G22000130001.

APPENDIX: SUPPLEMENTARY INFORMATION

1. Explicit expressions for p_{ij} and q_{ij}

The expressions for p_{ij} and q_{ij} are derived from the solution to the problem defined by Hamiltonian in Eq. (17), here recalled:

$$H(\mathcal{G}|\vec{\alpha}, \vec{\beta}) \equiv \sum_{i=1}^{N} \left[\alpha_i \sum_{t=1}^{T} \frac{k_i(t)}{T} + \beta_i \sum_{t=1}^{T} \frac{h_i(t)}{T} \right]. \tag{A1}$$

In Ref. [20] it is provided an exact formulation for these quantities. Specifically, through appropriate reparametrization, the model outlined by this Hamiltonian can be precisely mapped

to a combination of noninteracting one-dimensional Ising models. These models are then solvable analytically, with their solutions expressible in terms of a constant connection probability.

$$p_{ij} = \left(\frac{x_i x_j y_{ij} - 1}{2\sqrt{4x_i x_j + (x_i x_j y_{ij} - 1)^2}}\right)\left(\frac{(\lambda_{ij}^+)^T - (\lambda_{ij}^-)^T}{(\lambda_{ij}^+)^T + (\lambda_{ij}^-)^T}\right) + \frac{1}{2}, \tag{A2}$$

$$q_{ij}(\tau) = p_{ij}^2 + p_{ij}(1 - p_{ij}) \times \left(\frac{(\lambda_{ij}^-)^\tau (\lambda_{ij}^+)^{T-\tau} + (\lambda_{ij}^+)^\tau (\lambda_{ij}^-)^{T-\tau}}{(\lambda_{ij}^+)^T + (\lambda_{ij}^-)^T}\right), \tag{A3}$$

with

$$\lambda_{ij}^{\pm} = e^{J_{ij}} \cosh B_{ij} \pm \sqrt{e^{2J_{ij}} \sinh^2 B_{ij} + e^{-2J_{ij}}}, \tag{A4}$$

and

$$B_{ij} \equiv \tfrac{1}{2}[\ln(x_i) + \ln(x_j) + \ln(y_i) + \ln(y_i)]. \tag{A5}$$

$$J_{ij} \equiv \tfrac{1}{4}[\ln(y_i) + \ln(y_j)], \tag{A6}$$

where, $x_i = e^{-\alpha_i/T}$, while $y_i = e^{-\beta_i/T}$.

Based on these terms, we can determine the probability of observing or not observing a link at time $t + \tau$ given its presence or absence at time t. This result is then translated in the stochastic matrix defined in Eq. (21).

2. Autocovariance for the matrix B

Our main argument for computing the DT, when employing modularity that uses the persisting degree, is based on a heuristic that assumes after waiting τ^* [as defined in Eq. (28)], all matrices in the sequences $\mathcal{B}$ and $\mathcal{A}$ can be considered independent.

Using the argument in Ref. [47], we compute:

$$\mathrm{Cov}[B_{ij}(t), B_{ij}(t+\tau)] = \mathrm{Cov}[a_{ij}(t)a_{ij}(t+\tau), a_{ij}(t+\tau)a_{ij}(t+2\tau)], \tag{A7}$$

where the resulting calculation is as follows:

$$\begin{aligned}\mathrm{Cov}[B_{ij}(t), B_{ij}(t+\tau)] = E[a_{ij}(t)]^2[\mathrm{Cov}[a_{ij}(t), a_{ij}(t+\tau)]^2 + \mathrm{Var}[a_{ij}(t)] + \mathrm{Cov}[a_{ij}(t), a_{ij}(t+2\tau)]] \\ + \mathrm{Cov}[a_{ij}(t+\tau)]^2 + \mathrm{Var}[a_{ij}(t)]\mathrm{Cov}[a_{ij}(t), a_{ij}(t+2\tau)],\end{aligned} \tag{A8}$$

where we use the fact that $\mathrm{Cov}[a_{ij}(t), a_{ij}(t+\tau)] = \mathrm{Cov}[a_{ij}(t+\tau), a_{ij}(t+2\tau)]$ that, after substituting the values of the covariances as computed in Ref. [20] and the expected values, becomes:

$$\begin{aligned}\mathrm{Cov}[B_{ij}(t), B_{ij}(t+\tau)] &= p_{ij}^2[p_{ij}(1-p_{ij}) + 2g_{ij}(\tau) + g_{ij}(2\tau)] + g(\tau)^2 + p_{ij}(1-p_{ij})g_{ij}(2\tau) \\ &= p_{ij}^3(1-p_{ij}) + p_{ij}^2 g_{ij}(2\tau) + 2p_{ij}^2 g(\tau) + g_{ij}(\tau)^2 + p_{ij}(1-p_{ij})g_{ij}(2\tau).\end{aligned} \tag{A9}$$

Here, $g(\tau)$ is a function of τ, defined in Ref. [20], whose value decays with τ:

$$g(\tau) = p_{ij}(1-p_{ij})(\gamma)^\tau, \tag{A10}$$

with $\gamma < 1$. Considering Eq. (A9), we note that waiting τ^*, and considering the term $p_{ij}^3(1-p_{ij})$ small enough, it is equal to 0.

3. Generating community-preserving graph trajectories via Markov chains

Operationally, we begin by defining a network generated according to a stochastic block model characterized by an internal cluster connection probability, p_{in}, and an external connection probability, p_{out}. At this point, two different stochastic matrices are used to define the evolution of connections in this initial network. These matrices are characterized by p_{in} and q_{in} for internal cluster connections, and p_{out} and q_{out} for external connections. By varying the ratio between the densities of link and link persistence, we are able to study the behavior of our methodology. Trajectories are generated in a way that preserves both the network's average total degree $\bar{k}^*$ and the average total persistent degree $\bar{h}^*$.

The concept revolves around altering the ratio between the average contribution from internal cluster connections and that from external links. This goal is achieved by adjusting p_{in} and p_{out} as well as q_{in} and q_{out}. To maintain $\bar{k}^*$ and $\bar{h}^*$ as mentioned, we use the following formulas:

$$p_{\mathrm{in}}\frac{N}{n_{\mathrm{groups}}} + p_{\mathrm{out}}\frac{N(n_{\mathrm{groups}}-1)}{n_{\mathrm{groups}}} = \bar{k}^*, \tag{A11}$$

$$q_{\mathrm{in}}\frac{N}{n_{\mathrm{groups}}} + q_{\mathrm{out}}\frac{N(n_{\mathrm{groups}}-1)}{n_{\mathrm{groups}}} = \bar{h}^*, \tag{A12}$$

where n_{groups} represents the number of clusters, N is the total number of nodes in the network.

Hence, by defining p_{out} and q_{out} as functions of p_{in} and q_{in}, respectively:

$$p_{\mathrm{out}} = \epsilon p_{\mathrm{in}}, \tag{A13}$$

$$q_{\mathrm{out}} = \eta q_{\mathrm{in}}, \tag{A14}$$

where ϵ and η are parameters that determine the densities of internal and external links and persisting links within clusters, we can manipulate the differences between the internal and external clusters by adjusting these parameters.

Using this further conditions we obtain the following formulas:

$$\begin{aligned}p_{\mathrm{out}} &= \epsilon p_{\mathrm{in}} \\ p_{\mathrm{in}} &= \frac{n_{\mathrm{groups}}\bar{k}^*}{[1+\epsilon(n_{\mathrm{groups}}-1)]N}\end{aligned} \tag{A15}$$

$$q_{\text{out}} = \eta q_{\text{in}}$$
$$q_{\text{in}} = \frac{n_{\text{groups}}\bar{h}^*}{[1+\eta(n_{\text{groups}}-1)]N}. \tag{A16}$$

By using Eqs (A15) and (A16), we ensure that the expected value on the trajectory of $\langle \bar{h} \rangle = \bar{h}^*$ and $\langle \bar{k} \rangle = \bar{k}^*$.

The parameters η and ϵ can assume values within the following intervals:

$$\epsilon \in \{kv \mid k \in \mathbb{N},\ nv \leqslant 1\ \&\ nv > 0\}, \tag{A17}$$

$$\eta \in \{r\epsilon \mid r \in (0,1)\}. \tag{A18}$$

In this way, the network's evolution will preserve the condition whereby the persistent degree within the group and outside the group will always be lower, respectively, than the internal degree to the group and the external one. In the numerical simulations presented in Sec. V, we focused on a scenario where $n_{\text{groups}} = 2$.

4. Schematic presentation of the algorithm

Here we explain the steps followed to optimize the modularity function. The algorithm aims to maximize the modularity function by changing the labels associated with each element of the network. We exploit the method proposed by Clauset and Moore in Ref. [43], by readapting it to use the maximum entropy null models. In this sense we rewrite the modularity function in a general form, so that the function for which we want to find the maximum is

$$Q(g) = \sum_{i,j} \left[C_{ij} - P_{ij}^{(\text{null})}\right]\delta_{g(i)g(j)}, \tag{A19}$$

where depending on the features considered, C_{ij} is

$$C_{ij} = \begin{cases} a_{ij} & \text{static case} \\ \overline{a_{ij}} & \text{dynamic case no memory} \\ \overline{a_{ij}} + \overline{b_{ij}} & \text{dynamic case with memory} \end{cases}. \tag{A20}$$

While $P_{ij}^{(\text{null})}$ assumes the form of the null models presented in the main text. The implementation proposed in Ref. [43] works by defining a matrix ΔQ, whose elements are indicated with ΔQ_{vw} (defined below) for each couple of communities. The size of this matrix is initially assumed to be equal to the number of nodes, i.e., at each node is initially attached a unique label. The size is then reduced according to the merging procedure, so to reduce the number of communities. Hence, for our application, we define the initial value of ΔQ_{vw} as:

$$\Delta Q_{vw} = C_{vw} - P_{vw}^{(\text{null})} \quad \text{for} \quad j, i \in N. \tag{A21}$$

Then is selected the largest ΔQ_{vw}, and if it is positive, communities v and w are merged, the value of Q is incremented of ΔQ_{vw} and the matrix ΔQ is updated according to the following rule:

$$\Delta Q_{vk}^{\text{new}} = \Delta Q_{vk} + \Delta Q_{wk}. \tag{A22}$$

The procedure is repeated until it is no longer convenient to merge communities.

5. Change point detection in temporal community structures

Here, we revisit the structural break detection algorithm previously introduced in Ref. [20], with the aim of tracking communities' evolution over time. The central assumption of our approach is that there are periods during which the system's statistical properties, specifically community presence, remain static.

Our method is based on the binary segmentation technique [48]. In our proposal, we employ the Akaike information criterion (AIC) [49] as a cost function to measure uniformity among observations. The AIC, which considers both the likelihood of the model and the number of parameters, is defined as:

$$\text{AIC}_{\text{model}} = 2K_{\text{model}} - 2\ln(L_{\text{model}}) \tag{A23}$$

where K_{model} is the number of parameters, and L_{model} is the likelihood computed from the data.

Breaks in the structure are identified by solving the following:

$$t_{\text{break}} = \underset{t\in[1,\cdots,T-1]}{\text{argmin}}\ \Delta\text{AIC}(t), \tag{A24}$$

where $\Delta\text{AIC}(t)$, the difference in AIC before and after a potential break at time t, is calculated as:

$$\Delta\text{AIC}(t) = \text{AIC}_{\text{tot}} - \text{AIC}_{\text{diff}}(t), \tag{A25}$$

with

$$\text{AIC}_{\text{tot}} = 2K_{\text{model}} - 2\ln[L(\mathcal{G}_{0,T})], \tag{A26}$$

and

$$\text{AIC}_{\text{diff}}(t) = 4K_{\text{model}} - 2(\ln[L(\mathcal{G}_{0,t})] + \ln[L(\mathcal{G}_{t,T})]). \tag{A27}$$

A break t_{break} is valid if $\Delta\text{AIC}(t_{\text{break}}) < 0$. Otherwise, it is assumed that no structural breaks are present. Once a structural break is determined, we can isolate the segment $[0, t_{\text{break}}]$ and repeat the analysis on it. This iterative process continues until no further significant breaks are detected. The final break is marked as $t_{\text{break final}}$. After identifying this break, the segment $[0, t_{\text{break final}}]$ is removed, and the focus shifts to the remaining segment $[t_{\text{break final}}, T]$. This approach results in a collection of segments that, according to the model, are generated under conditions of relatively constant parameters within each segment. In our memory-inclusive model, the parameters count K_{model} equals $2N$, where N is the number of nodes. The log likelihood is given by:

$$\ln[L(\mathcal{G})] = -H - \ln(Z), \tag{A28}$$

where H is defined in Eq. (17), and $\ln Z$, derived in Ref. [20], is calculated as:

$$\ln Z = \sum_{i<j} \ln[(\lambda_{ij}^{+})^T + (\lambda_{ij}^{-})^T], \tag{A29}$$

with $\lambda_{ij}^{\pm}$ as defined in Eq. (A4). For further details, see Ref. [20].

To test this approach, we generated a synthetic data set using a block model characterized by probabilities typical of community structures. The system is designed to evolve according to matrices Q and P, with elements q_{ij} and p_{ij} respectively. These matrices are structured to maintain a higher

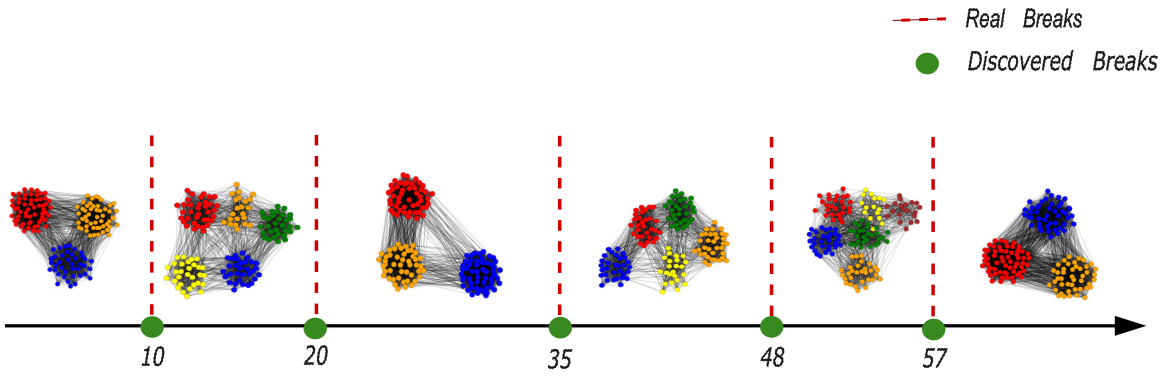

FIG. 9. A representation of the evolution of synthetically generated temporal networks is shown. The graph trajectory includes 150 nodes. Along the x axis, the real synthetic breaks are depicted (red vertical lines), and each defined segment illustrates the related average network, highlighting the real community structure. Green dots represent the discovered structural breaks, which perfectly align with the real ones.

density of intracommunity links and persisting links compared to intercommunity links. After generating an initial segment of a specified length, we added a new segment with the same nodes but altered the community structure and introduced noise by randomly reshuffling some node labels. The experiment was conducted using a synthetic trajectory of 75 snapshots, each containing a network of 150 nodes.

As illustrated in Fig. 9, our procedure effectively tracks the evolution of communities. Utilizing the proposed modularity formula [Eq. (25)] in conjunction with our method for detecting change points enhances its applicability to networks where nodes change membership over time but remain stationary within sufficiently long segments. In our synthetic experiment, we successfully captured all structural breaks. Although the segments are accurately identified, this does not necessarily guarantee perfect alignment with the underlying changes in the DGP, which are driven by the evolution of the hidden community structure.

The results, as detailed in Table I, reveal some discrepancies in the communities detected across different segments. In some segments, the identified communities do not perfectly match the actual ones. A higher number of communities and varying degrees of connectivity between and within these communities complicate the task of accurately capturing community structures, especially when compared to simpler cases in other segments.

6. Analyzing the evolution of detected communities and the role of memory

Here we show how, if memory is encoded in the DGP, selecting the appropriate time window for data aggregation correlates with the observed memory in the data, as measured by the normalized autocovariance. We display how this selection may be relevant in enhancing the algorithm's ability to detect communities. We carried out a series of numerical experiments to verify whether, in line with the findings from Sec. VI, the point at which the normalized autocovariance peaks corresponds to a more discernible community structure as detected by the memory-exploiting modularity (25). In our simulations, we start with a time-varying graph generated as described in Sec. A 3, featuring a structure of five communities. From this graph, we create additional structures by disaggregating and aggregating snapshots. The disaggregation process involves dividing each snapshot from the original time-varying graph into a number of subsnapshots. Each subsnapshot G_{sub} comprises links randomly selected from the original snapshot, ensuring that the union of all subgraphs reconstructs the original, hence:

$$\bigcup_{t=1}^{n} G_{\text{sub}}(t) = G^{\text{original}}, \tag{A30}$$

TABLE I. ARI for different segments identified in synthetic trajectories.

Segment	0–10	10–20	20–35	35–48	48–57	57–75
ARI	1.00	0.81	1.00	0.83	0.69	1.00

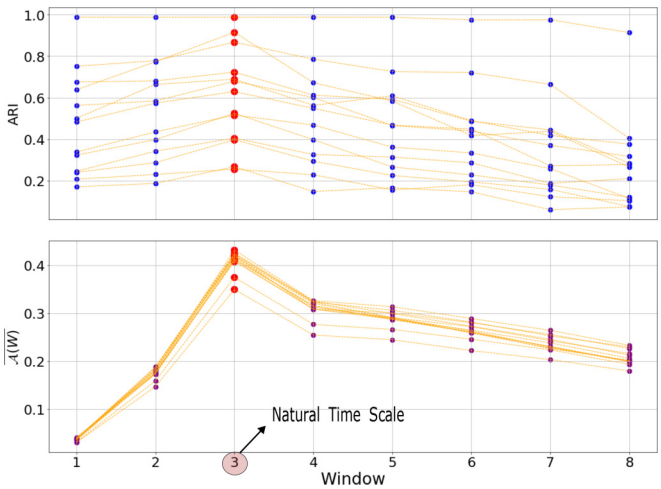

FIG. 10. Results from the numerical simulations described in Appendix 6. Each line is related to different initial values for Δk and Δh at the natural time scale, which is consistently 3 for all trajectories. Each network in the simulations comprises 200 nodes, 15 time steps, and five communities. Red dots represent the maximum for the given metric regarding the aggregation level.

where the operator $\bigcup$ operates in such a way that if a link is present in both snapshot t_1 and snapshot t_2, it is counted only once in the aggregated graph.

Temporal order is maintained; for example, if the first snapshot is divided into n graphs, these will occupy the same position as the original snapshot in the complete time-varying graph. The aggregation process functions by summing multiple snapshots to form a new snapshot, which will contain all the links from the combined structures. The procedure ensures that there will be nonoverlapping snapshots. Through this approach, we establish a controlled setting with a natural scale at which memory is incorporated into the generative process. This enables us to numerically check whether this scale corresponds to the point where the normalized autocovariance is higher and where the modularity, with known ground-truth communities, performs optimally.

In Fig. 10 are shown the results from a series of numerical simulations for different values of Δk and Δh in a temporal network featuring five ground-truth communities. Each point pertains to an aggregation state of the network, selecting progressively more aggregated temporal windows as the value on the x axis increases. The two charts display the ARI value (top) for each aggregation window and the value of normalized autocovariance (bottom). Points connected by the same line represent the same network across different aggregation windows. The points highlighted in red correspond to the peak aggregation window for the variable of interest. The natural scale of aggregation is 3, which is the baseline from which all other temporal networks were generated. The natural time scale corresponds to the maximum for both metrics, supporting our initial hypothesis.

[1] O. E. Williams, L. Lacasa, A. P. Millán, and V. Latora, The shape of memory in temporal networks, Nature Commun. **13**, 499 (2022).

[2] F. Bauzá Mingueza, M. Floría, J. Gómez-Gardeñes, A. Arenas, and A. Cardillo, Characterization of interactions' persistence in time-varying networks, Sci. Rep. **13**, 765 (2023).

[3] N. Pedreschi, D. Battaglia, and A. Barrat, The temporal rich club phenomenon, Nature Phys. **18**, 931 (2022).

[4] S. N. Chowdhury, S. Majhi, D. Ghosh, and A. Prasad, Convergence of chaotic attractors due to interaction based on closeness, Phys. Lett. A **383**, 125997 (2019).

[5] S. N. Chowdhury, S. Majhi, M. Ozer, D. Ghosh, and M. Perc, Synchronization to extreme events in moving agents, New J. Phys. **21**, 073048 (2019).

[6] V. Kohar, P. Ji, A. Choudhary, S. Sinha, and J. Kurths, Synchronization in time-varying networks, Phys. Rev. E **90**, 022812 (2014).

[7] S. N. Chowdhury, S. Majhi, and D. Ghosh, Distance dependent competitive interactions in a frustrated network of mobile agents, IEEE Trans. Network Sci. Eng. **7**, 3159 (2020).

[8] G. Mikaberidze, S. N. Chowdhury, A. Hastings, and R. M. D'Souza, Consensus formation among mobile agents in networks of heterogeneous interaction venues, Chaos, Solitons Fractals **178**, 114298 (2024).

[9] K. Xu, Stochastic block transition models for dynamic networks, in *Artificial Intelligence and Statistics* (PMLR, 2015), pp. 1079–1087.

[10] K. Ide, A. Namatame, L. Ponnambalam, F. Xiuju, and R. S. M. Goh, A study on relationship between modularity and diffusion dynamics in networks from spectral analysis perspective, Int. J. Adv. Comput. Sci. Appl. **5** (2014).

[11] R. M. May, Will a large complex system be stable? Nature (London) **238**, 413 (1972).

[12] F. de Castro, S. M. Adl, S. Allesina, R. D. Bardgett, T. Bolger, J. J. Dalzell, M. Emmerson, T. Fleming, D. Garlaschelli, J. Grilli *et al.*, Local stability properties of complex, species-rich soil food webs with functional block structure, Ecol. Evol. **11**, 16070 (2021).

[13] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E **69**, 026113 (2004).

[14] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications, Phys. Rev. E **84**, 066106 (2011).

[15] M. Rosvall and C. T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. **105**, 1118 (2008).

[16] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, Community structure in time-dependent, multiscale, and multiplex networks, Science **328**, 876 (2010).

[17] P. Barucca, F. Lillo, P. Mazzarisi, and D. Tantari, Disentangling group and link persistence in dynamic stochastic block models, J. Stat. Mech. (2018) 123407.

[18] X. Zhang, C. Moore, and M. E. Newman, Random graph models for dynamic networks, Eur. Phys. J. B **90**, 1 (2017).

[19] T. Squartini and D. Garlaschelli, *Maximum-Entropy Networks: Pattern Detection, Network Reconstruction and Graph Combinatorics* (Springer, Berlin, 2017).

[20] G. V. Clemente, C. J. Tessone, and D. Garlaschelli, Temporal networks with node-specific memory: unbiased inference of transition probabilities, relaxation times and structural breaks, arXiv:2311.16981.

[21] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová, Inference and phase transitions in the detection of modules in sparse networks, Phys. Rev. Lett. **107**, 065701 (2011).

[22] A. Ghasemian, P. Zhang, A. Clauset, C. Moore, and L. Peel, Detectability thresholds and optimal algorithms for community structure in dynamic networks, Phys. Rev. X **6**, 031005 (2016).

[23] J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.-F. Pinton, M. Quaggiotto, W. Van den Broeck, C. Régis, B. Lina *et al.*, High-resolution measurements of face-to-face contact patterns in a primary school, PloS One **6**, e23176 (2011).

[24] S. Fortunato, Community detection in graphs, Phys. Rep. **486**, 75 (2010).

[25] N. J. Gotelli and G. R. Graves, *Null models in ecology* (Smithsonian Institute, Washington, DC, 1996).

[26] M. E. J. Newman, Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E **74**, 036104 (2006).

[27] T. Squartini, R. Mastrandrea, and D. Garlaschelli, Unbiased sampling of network ensembles, New J. Phys. **17**, 023052 (2015).

[28] S. Maslov, K. Sneppen, and A. Zaliznyak, Detection of topological patterns in complex networks: Correlation profile of the internet, Physica A **333**, 529 (2004).

[29] E. T. Jaynes, Information theory and statistical mechanics, Phys. Rev. **106**, 620 (1957).

[30] T. Caruso, G. V. Clemente, M. C. Rillig, and D. Garlaschelli, Fluctuating ecological networks: A synthesis of maximum-entropy approaches for pattern detection and process inference, Methods Ecol. Evol. **13**, 2306 (2022).

[31] J. Park and M. E. J. Newman, Statistical mechanics of networks, Phys. Rev. E **70**, 066117 (2004).

[32] D. Garlaschelli and M. I. Loffredo, Maximum likelihood: Extracting unbiased information from complex networks, Phys. Rev. E **78**, 015101(R) (2008).

[33] J. Reichardt and M. Leone, (Un) detectable cluster structure in sparse networks, Phys. Rev. Lett. **101**, 078701 (2008).

[34] R. R. Nadakuditi and M. E. J. Newman, Graph spectra and the detectability of community structure in networks, Phys. Rev. Lett. **108**, 188701 (2012).

[35] P. W. Holland, K. B. Laskey, and S. Leinhardt, Stochastic blockmodels: First steps, Social Networks **5**, 109 (1983).

[36] E. Mossel, J. Neeman, and A. Sly, Reconstruction and estimation in the planted partition model, Probab. Theory Relat. Fields **162**, 431 (2015).

[37] L. Hubert and P. Arabie, Comparing partitions, J. Classification **2**, 193 (1985).

[38] M. E. J. Newman, Spectral methods for community detection and graph partitioning, Phys. Rev. E **88**, 042822 (2013).

[39] J.-G. Young, P. Desrosiers, L. Hébert-Dufresne, E. Laurence, and L. J. Dubé, Finite-size analysis of the detectability limit of the stochastic block model, Phys. Rev. E **95**, 062304 (2017).

[40] G. Rossetti and R. Cazabet, Community discovery in dynamic networks: A survey, ACM Computing Surveys (CSUR) **51**, 1 (2019).

[41] G. Palla, A.-L. Barabási, and T. Vicsek, Quantifying social group evolution, Nature (London) **446**, 664 (2007).

[42] L. Dall'Amico, R. Couillet, and N. Tremblay, Community detection in sparse time-evolving graphs with a dynamical Bethe-Hessian, *Advances in Neural Information Processing Systems*, edited by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Vol. 33 (Curran Associates, Inc., 2020), pp. 7486–7497.

[43] A. Clauset, M. E. J. Newman, and C. Moore, Finding community structure in very large networks, Phys. Rev. E **70**, 066111 (2004).

[44] G. Krings, M. Karsai, S. Bernhardsson, V. D. Blondel, and J. Saramäki, Effects of time window size and placement on the structure of an aggregated communication network, EPJ Data Science **1**, 4 (2012).

[45] A. Clauset and N. Eagle, Persistence and periodicity in a dynamic proximity network, in *Proceedings of the DIMACS/DyDAn Workshop on Computational Methods for Dynamic Interaction Networks* (IEEE, Piscataway, NJ, 2007).

[46] D. S. Bassett, M. A. Porter, N. F. Wymbs, S. T. Grafton, J. M. Carlson, and P. J. Mucha, Robust detection of dynamic community structure in networks, Chaos **23**, 013142 (2013).

[47] G. W. Bohrnstedt and A. S. Goldberger, On the exact covariance of products of random variables, J. Am. Stat. Assoc. **64**, 1439 (1969).

[48] A. Scott and M. Knott, A cluster analysis method for grouping means in the analysis of variance, Biometrics **30**, 507 (1974).

[49] H. Akaike, A new look at the statistical model identification, IEEE Trans. Autom. Control **19**, 716 (1974).