



**HAL**  
open science

# Longitudinal Modularity, a Modularity for Link Streams

Victor Brabant, Yasaman Asgari, Pierre Borgnat, Angela Bonifati, Remy  
Cazabet

► **To cite this version:**

Victor Brabant, Yasaman Asgari, Pierre Borgnat, Angela Bonifati, Remy Cazabet. Longitudinal Modularity, a Modularity for Link Streams. 2024. hal-04731945

**HAL Id: hal-04731945**

**<https://hal.science/hal-04731945v1>**

Preprint submitted on 11 Oct 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Longitudinal Modularity, a Modularity for Link Streams

Victor Brabant<sup>1</sup>, Yasaman Asgari<sup>2</sup>, Pierre Borgnat<sup>3</sup>, Angela Bonifati<sup>1</sup>, and  
Rémy Cazabet<sup>1</sup>

1. UCBL, CNRS, INSA Lyon, LIRIS, UMR5205, F-69622 Villeurbanne, France
2. Department of Mathematical Modeling and Machine Learning, Digital Society Initiative (DSI), University of Zurich, CH-8057 Zurich, Switzerland
3. CNRS, Univ de Lyon, ENSL, Laboratoire de Physique, F-69342 Lyon, France  
victorbrabant@gmail.com

**Abstract.** Temporal networks are commonly used to model real-life phenomena. When these phenomena represent interactions and are captured at a fine-grained temporal resolution, they are modeled as link streams. Community detection is an essential network analysis task. Although many methods exist for static networks, and some methods have been developed for temporal networks represented as sequences of snapshots, few works can handle link streams. This article introduces the first adaptation of the well-known Modularity quality function to link streams. Unlike existing methods, it is independent of the time scale of analysis. After introducing the quality function, and its relation to existing static and dynamic definitions of Modularity, we show experimentally its relevance for dynamic community evaluation.

## 1 Introduction

Complex networks are powerful tools for modeling real-life phenomena, such as social interactions, economic transactions, and biological interactions. A classic problem in this field is discovering the community structure of a given network. While there is an extensive body of literature addressing this challenge [16], most of it focuses on static networks. Temporal or dynamic networks are frequently found when representing real-life phenomena, and discovering communities in such data represent a challenge of its own. Various methods have already been proposed to handle temporal networks [32], however many of them are only able to work on slowly evolving graphs [13], i.e., graphs that can be represented as sequences of well-behaved static networks. Many real-world phenomena are instead composed of a flow of interactions occurring at a fast scale, that cannot be interpreted as a usual static network at any single point in time. Confronted to such data, the usual approach in the literature is to aggregate the data into sequences of snapshots, using a time-window. However, this approach raises a number of difficulties, such as the choice of an appropriate time-window, and the loss of temporal information resulting from the aggregation. Instead, such data is better modeled as *stream graphs*, or *link streams* [20], frameworks designed to

directly manipulate such fine-grained temporal data, without any unnecessary aggregation. Many common network analysis problems have been redefined and adapted to the link stream case, such as clustering coefficient, connected components, or shortest paths[20], cliques as  $\Delta$ -cliques[37], random walk centralities [4], etc.

A few methods in the literature have proposed to detect communities in link streams, however those methods are limited to detect some specific subcase of partitions, such as non-evolving communities [25] or having only one step of evolution and requiring a temporal scale of analysis[8]. Instead, we propose in this work to adapt a well-known quality function for community detection in static networks, the Modularity. Although an adaptation of Modularity for sequences of snapshots exists[26], it cannot work on link streams. In this article, we propose the first generalization of Modularity for link streams, that we name Longitudinal Modularity (L-Modularity).

Section 2 defines link streams and introduce notations we will use throughout the paper. It also defines the type of dynamic communities we are considering. Section 3 provides an overview of related works and their limits. Section 4 introduces the definition of the Longitudinal Modularity we propose. Section 5 introduces a set of properties one should expect a good temporal community definition to respect, and show that L-Modularity does indeed respect those properties. Section 6 demonstrate through experimentations the relevance of our quality function. Finally, we conclude in Section 7.

## 2 Definitions: Link Stream and Temporal Community

Temporal networks are known under many names and formalisms, such as temporal networks, dynamic networks, time varying networks, etc. Similarly, various concepts of temporal communities are used in the literature. In this section, we clarify the definition of link streams, and what makes this type of representation relevant to study, and similarly for the notion of temporal community.

### 2.1 Link Streams

**Slowly Evolving Graphs and Link Streams** Although many formalisms exist to represent temporal networks, in this article we make a fundamental distinction between two distinct types: slowly evolving graphs (SEG) and Link streams. SEG [13] correspond to networks that can be seen as series of static graphs, or as a network evolving *edge by edge*, while remaining at every time a conventional, well-behaved network, that can be studied with the tools of network science. Among networks being typical SEG, one can mention friendship in social networks, or yearly snapshots composed by aggregating interactions among users of an online platform. Multiple community detection methods have been developed to deal with such networks [32], e.g., those compared experimentally in [13].

SEGs are opposed to link streams, that correspond to the original form of many real data, in which interactions are simply collected as events  $(uv, t)$ , corresponding to an *interaction* between nodes  $uv$  at time  $t$ , e.g., a physical interaction, an instantaneous message by phone or on a social platform, etc. In such data, there is no well-behaved network at each point in time, thus the usual approach consists in first aggregating the data using sliding windows, to obtain a sequence of usual static graphs. But this approach has many drawbacks, from the arbitrary choice of the duration of the time window, to the loss of temporal details inside the aggregated period, or the artifacts introduced by abrupt change at their arbitrary boundaries. Thus a recent trend of research (notably, stemming from [20]) consists in designing concepts and methods able to work directly on those objects —called here *link streams*— without having to use aggregation periods.

### Formal definition

**Definition 1.** A *link stream*  $\mathcal{L}$  is defined by a triplet  $(T, V, E)$  where  $T \subset \mathbb{R}$  is a time interval,  $V$  a finite set of  $N \in \mathbb{N}$  nodes, and  $E = \{(uv, t) \in V^2 \times T\}$  a finite set of interactions.

In the following we focus on the case of simple dynamic graphs where interactions are instantaneous, undirected, and unweighted. Although this does not affect our contribution,  $T$  is considered to be discrete. To introduce the notations used in the following (see also Fig. 1):

- Let  $uv_t = 1$  if  $(uv, t) \in E$ , else 0.
- For a subset of time  $T' \subset T$ , define  $L_{uv, T'} = \sum_{t \in T'} uv_t$  as the number of interactions between nodes  $u$  and  $v$  over  $T'$ .
- Let  $L_{uv} = L_{uv, T}$  denote the total number of interactions between nodes  $u$  and  $v$ .
- Similar to static graphs, let  $k_u = \sum_{v \in V} L_{uv}$  denote the degree of a node  $u$  and  $m = \sum_{u \in V} k_u / 2 = |E|$  denote the total number of interactions in the link stream.

## 2.2 Temporal Communities

As there are multiple ways to define a temporal network, there are also multiple ways to define temporal communities. In static networks, communities are usually defined as complete partitions, i.e., a set of set of nodes, such as each node belongs to exactly one community. In this article, we retain the principle of non-overlapping communities, but require 1) communities to be able to evolve with time, i.e., a community is a set of *node-time* pairs  $(u, t)$ , 2) nodes to be able to belong to no community over some periods. Indeed nodes might have long periods of inactivity (e.g., nights in a fine-scale dataset, or even nodes that have left the systems, without us having this information, e.g., a phone number that is no longer attributed), and it would make little sense to try to include those inactive nodes in another partition, much as it would not make sense to include a node without edges in a static partition.

**Definition 2.** A dynamic community structure  $\mathcal{C}$  over a link stream  $L = (T, V, E)$  is a set of non-empty and mutually exclusive communities composed of sets of node-time pairs  $\{(u_1, t_1), (u_1, t_2), \dots, (u_2, t_3), (u_2, t_4), \dots\}$ .

To introduce the notations used in the following:

- $T_{u \in C} = \{t \in T \text{ s.t. } (u, t) \in C\}$  represents the times when a node  $u$  belongs to community  $C$ .
- $T_{uv \in C} = T_{u \in C} \cap T_{v \in C}$  denotes the times nodes  $u$  and  $v$  simultaneously belong to community  $C$ .
- $L_{uv \in C} = L_{uv, T_{uv \in C}}$  represents the number of interactions between nodes  $u$  and  $v$  within community  $C$ .
- $T_C = \bigcup_{u \in V} T_{u \in C}$  denotes the existence time of community  $C$ .
- $C_u = \{C \in \mathcal{C} \text{ s. t. } T_{u \in C} \neq \emptyset\}$  denotes the set of communities visited by node  $u$ .

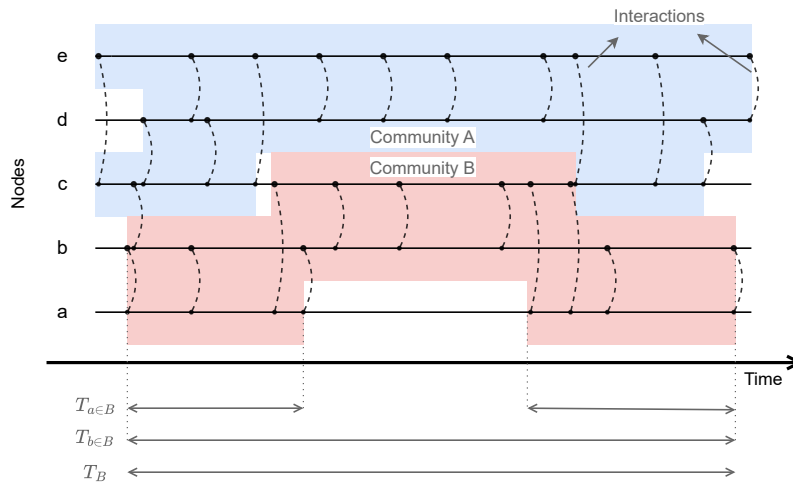


Fig. 1: Illustration of a link stream with a community structure depicted in blue and red. The figure includes representations of temporal community notations. The total interactions between nodes  $b$  and  $c$  are quantified as  $L_{bc} = 4$ . Within community  $B$ , these interactions total  $L_{bc \in B} = 3$ . Additionally, the degree of node  $a$  is represented as  $k_a = 8$ .

### 3 Related work

Detecting communities in temporal networks has been the focus of many previous research. However, most of these works are based on the so-called *Instant-*

*optimal* approach[32], i.e., first searching for static communities in a succession of snapshots, and then matching the partitions found using different strategies. The originality of our approach is that 1) We deal directly with link-stream data, without the need of aggregating into snapshots, and 2) We provide a quantitative definition of dynamic communities, that can be used as a quality function to evaluate a partition of a link stream. In this section, we will first discuss methods having only the first property, then those having only the second. Finally, we will see that the few methods at the intersection between the two suffer from important limitations.

### 3.1 Community Detection on Stream Graphs, without objective function

**Progressively Evolving Networks** A few methods in the literature have proposed to get rid of snapshots by considering network evolution at the finest temporal granularity, updating communities at each network modification, such as node/edge addition/removal, e.g., [11,33,6]. However, these methods are working on a particular type of temporal graphs, Progressively Evolving Graphs[13], in which a well-behaved network exists at each point in time. This network is simply updated by adding or removing elements, but remains observable at any point in time, in a form on which static methods can be applied on it. This type of network corresponds for instance to *relational data* such as friendships in a social media, or physical connections between routers in the internet. These methods are thus unable to deal with link streams, representing interactional data such as email, phone call, face-to-face interactions or messages in social media, in which the network exists only when considering interactions occurring at different times.

**Link Streams** A few methods have been proposed to search for persistent groups of nodes in Link Streams. In [37], the authors propose to discover  $\Delta$  cliques, i.e., groups of nodes that interact with each other at least once over a period  $\Delta$ . In [7], the authors search for groups of nodes that represent a consistently *good* group—in terms of *conductance*— over a sufficiently large period of time.

### 3.2 Community Detection on Snapshot Sequences, optimizing a global objective function

Modern static methods for community detection mostly rely on a quantitative definition of what good communities are, expressed as an objective function, e.g., Modularity[27], information compression on random walks[35] or SBM inference [29]. Once this objective is provided, one can search for the partition optimizing it, usually through greedy heuristics, given the complexity of the problem. Methods in the previous section instead rely on ad-hoc rules, or the discovery of predefined patterns, such as cliques. The limit of these approaches is that they

are not able to compare two partitions: they are only able to find 1) the only valid partition (pattern mining), or 2) a single partition as the result of a process.

Two approaches have proposed to adapt Modularity for temporal networks, i.e., having a single Modularity score for a temporal partition, thus explicitly incorporating the smoothing issue into their frameworks: the Average Modularity (A-Modularity) [2] and the Multislice Modularity (MS-Modularity) [26].

**Average Modularity** is defined only on a single, stationary partition of the temporal network, i.e., nodes are not allowed to join or leave communities along time. It is simply defined as the average modularity of the partition over all snapshots representing the graph.

**Multislice Modularity** is a more expressive adaptation of modularity, allowing nodes to change affiliation from one snapshot to the next. Its principle is to create a single multislice graph from a sequence of snapshots by adding artificial edges, known as interslice coupling, between successive temporal instances of each node. The authors introduce their formula for Multislice Modularity from the perspective of the flow stability of communities under a Laplacian dynamic [14]. Noting  $A_{ijs}$  the adjacency of nodes  $i$  and  $j$  at slice  $s$ ,  $k_{is}$  the degree of node  $i$  at slice  $s$ ,  $m_s$  the number of edges in slice  $s$ ,  $\omega_{irs}$  the interslice weight between instances of slices  $s$  and  $r$  of node  $i$ , and  $2\mu = \sum_s 2m_s + \sum_i \sum_{r,s} \omega_{irs}$ , MS-modularity is:

$$Q = \frac{1}{2\mu} \sum_{C \in \mathcal{C}} \sum_{i,j \in V^2} \sum_{r,s \in S^2} \left[ \left( A_{ijs} - \frac{k_{is}k_{js}}{2m_s} \right) \delta_{sr} + \omega_{irs} \delta_{ij} \right] \delta_{ir \in C} \delta_{js \in C} \quad (1)$$

MS-Modularity represents a convincing solution for temporal graphs provided as sequences of snapshots. However, as we will discuss when proposing our approach, it suffers from two limitations: 1) It works only on Progressively Evolving Graphs, and not on Link Streams, thus requiring to use snapshot aggregation as a preprocessing step, and 2) The results are highly dependent on the timescale chosen for aggregating into snapshots.

Beyond modularity, a few methods have been proposed to adapt the SBM inference approach to dynamic settings, such as [38]. However, those methods are usually very costly to optimize, and cannot be adapted to link streams. A similar remark can be done for methods based on tensor decomposition, notably using the NMF [23]. Their complexity becomes quickly intractable as the number of steps considered increases.

### 3.3 Link streams and Objective Functions

To the best of our knowledge, no version of the Modularity defined on link streams exists in the literature. At least one work has been conducted on SBM approaches [25], by considering that the probability of observing interactions between groups is modeled by a function of time. However, the authors show that

to ensure identifiability, the partition must be stable, i.e., nodes cannot change communities. Another related work has been introduced in [8], based on the principle of flow stability. However, the method requires to choose a resolution parameter, and provide only two sets of partitions, the initial and final ones.

Contrary to these methods, in this article we introduce L-Modularity, an adaptation of Modularity to link streams, allowing to assign a score for any partition of nodes of a link stream, including nodes changing communities, or even having no affiliation during some periods.

## 4 Longitudinal modularity

Modularity [27] is one of the most widely used methods for analyzing community structures in networks. Despite its known limitations, it is widely used for its intuitive definition, straightforward quality function, and ease of optimization. This simplicity makes it the ideal candidate for a first approach towards a quantitative definition of communities in link-streams.

Modularity of a community structure  $\mathcal{C}$  over a network is based on the comparison of two terms: the observed number  $\mathbb{L}_{\mathcal{C}}$  of edges within communities and their expected numbers  $\mathbb{E}_{\mathcal{C}}$  based on a random null model. We can therefore express Modularity in a generic form as:

$$Q = \mathbb{L}_{\mathcal{C}} - \mathbb{E}_{\mathcal{C}} \quad (2)$$

The higher the Modularity, the more exceptional is the density inside communities, given the reference null model. Modularity was first introduced using the configuration model as reference [27], which rewires edges while preserving the degrees. Denoting  $A$  the adjacency matrix of an unweighted and undirected network,  $k_i$  the degree of node  $i$ ,  $2m = \sum_i k_i$  twice the number of edges in the network, and  $\mathcal{C}$  a community structure over it, the Modularity according to the configuration model is:

$$Q = \frac{1}{2m} \sum_{\mathcal{C} \in \mathcal{C}} \sum_{i,j \in V^2} \left[ A_{ij} - \frac{k_i k_j}{2m} \right] \delta_{i \in \mathcal{C}} \delta_{j \in \mathcal{C}} \quad (3)$$

The selection of an appropriate null model is essential for defining modularity. Several null models have been proposed for static networks [9,12], and significant efforts have been made to adapt Modularity for different types of networks. This includes weighted and directed networks [1], bipartite networks [3], multi-layer networks [28], and hypergraphs [30]. As mentioned in section 3.2, Modularity has also been adapted for multislice and temporal networks, in the context of sequence of snapshots[26]. In the remaining of this section, we will show how we can formalize a generic form of temporal Modularity, as was done in Equation 2 for the static one. We will then propose an instantiation of this generic formula for link streams, L-Modularity.



#### 4.1 Abstract temporal modularity

Multislice Modularity (later, MS-Modularity) was originally expressed (Eq. 1) in a way which is specific to 1) a representation of dynamic graphs —sequences of snapshots 2) a specific null model —preserving degrees in each snapshots— and 3) a particular way to ensure stable communities —inter-snapshot edges with tunable weights.

We observe that, as we have done with static Modularity in 2, the general principle of a dynamic Modularity can be expressed in an abstract way as: the observed number of edges within communities  $\mathbb{L}_C$ , minus their expected number  $\mathbb{E}_C$  based on a null model, plus a smoothness term  $\mathbb{S}_C$ , penalizing nodes changing communities:

$$Q = \mathbb{L}_C - \mathbb{E}_C - \mathbb{S}_C \quad (4)$$

In MS-Modularity (Eq. 1),  $\mathbb{L}_C$  comes from the sum over edges present inside communities in each snapshot ( $A_{ijs}$ ), the expected edges correspond to the expected number of edges in each snapshot given their degrees in that snapshot ( $\frac{k_{is}k_{js}}{2m_s}$ ), and the smoothness term comes from inter-snapshot edges not being inside communities ( $\omega_{irs}$ ).

We observe however that this particular instantiation of the abstract temporal Modularity of Eq. 4 cannot work for link streams: since a valid graph does not exist at any given time  $t$ , it is not relevant to use the instantaneous degree, nor to represent a smoothness term as inter-temporal edges. Only  $\mathbb{L}_C$ , i.e., the observed number of edges inside communities, remain a valid notion in link streams. As a consequence, we will in the following introduce instantiations of the expected number of edges  $\mathbb{E}_C$  and of the smoothness term  $\mathbb{S}_C$  tailored for link streams.

#### 4.2 Internal Edges Count $\mathbb{L}_C$

Given our definitions of link streams and communities (Sec. 2), counting the number of edges inside communities is as simple as in static networks: an interaction  $uv_t$  belongs to a community  $C$  if both  $u$  and  $v$  belongs to  $C$  at time  $t$ . Since we defined  $L_{uv \in C}$  the number of interactions between  $u$  and  $v$  in community  $C$ , the total number of internal interactions can be computed as:

$$\mathbb{L}_C = \sum_{u,v \in V^2} \sum_{C \in \mathcal{C}} L_{uv \in C} \quad (5)$$

#### 4.3 Link Expectation Null Model

In static Modularity, the most referenced null model is the configuration model, which randomizes the edges while preserving the degree sequence of the nodes [27]. With the notations of Eq. 3, the expectation of the presence of an edge

between nodes  $i$  and  $j$  in a community  $C \in \mathcal{C}$ , according to this null model is given by

$$\mathbb{E}[A_{ij}] = \frac{k_i k_j}{2m} \delta_{i \in C} \delta_{j \in C} \quad (6)$$

In other words, two nodes with high degrees are more likely to have a link between them. Conversely, observing a link between two nodes with low degrees is unexpected and may indicate the presence of a hidden community structure.

The expectation term of MS-Modularity in the sense of the abstract temporal modularity (Eq. 4) is a function of the degrees of the nodes in each snapshot. With the notations of Eq. 1, the expected number of links between nodes  $i$  and  $j$  in a community  $C$  over a set of snapshots  $S$  is

$$\mathbb{E} \left[ \sum_{s \in S} A_{ijs} \right] \propto \sum_{s \in S} \frac{k_{is} k_{js}}{2m_s} \delta_{is \in C} \delta_{js \in C} \quad (7)$$

In other words, the expected number of links is proportional to the sum of the expected numbers of links in each snapshot, with each snapshot's expectations determined by its specific configuration model. This approach is referred to as the SnapS null model in Gauvin et al. (2022) [17].

In the context of link streams, we consider the null model denoted as  $p[k, E]$  in Gauvin et al. (2022) [17]. This null model preserves the total degree of each node, but not the temporal sequence of interactions. For convenience, we name this null model the *longitudinal configuration model*, as it directly generalizes the configuration model used for static networks. Note that this null model can be used both for link streams or snapshots.

Compared with the null model used in MS-Modularity, it seems to better take into account the temporal nature of the data: if we imagine a temporal network in which a pair of nodes interact actively over short periods, while staying inactive during others, then using the Longitudinal Configuration Model as reference, we will naturally consider these periods of activity as exceptional. Instead, using MS-Modularity null model, the activity in each period is compared only with other nodes activity in that same period, and not with the activity of the same nodes in other periods.

The modularity expectation terms are typically derived directly from the null model, corresponding to the stationary probability of two random walkers arriving at each pair of nodes based on the transition probability given by the null model [14]. If we choose this approach, we must define transition probabilities between time instances for each node, thereby introducing artificial edges, which is precisely what we seek to avoid. We propose an alternative method, termed the *longitudinal expectation* approach. This approach constructs expectation terms based on the total duration of the link stream and multiplies them with temporal weights defined by node presence in communities. We present three versions of the expected number of interactions between nodes  $u$  and  $v$  in a community  $C$ .

**The co-membership expectation** ( $\mathbb{E}_{CM}$ ) (Eq. 8) represents the straightforward intuition that two nodes can only interact in time ranges in which they are both present.

$$\mathbb{E}_{CM} [L_{uv \in C}] = \frac{k_u k_v}{2m} \frac{|T_{uv \in C}|}{|T|} \quad (8)$$

**The joint membership expectation** ( $\mathbb{E}_{JM}$ ) (Eq. 9), considers the overall structure of the community in a way that promotes stationary or nearly stationary communities, where nodes remain members for the entire duration of the community’s existence. One could assume that, just as a ”perfect” community in a static network is a clique disconnected from the rest of the graph, a ”perfect” community in a link stream is a set of nodes with similar properties that remain unchanged—i.e., no node additions or removals—during its existence.

$$\mathbb{E}_{JM} [L_{uv \in C}] = \frac{k_u k_v}{2m} \frac{|T_C|}{|T|} \text{ if } C \in C_u \cap C_v, \text{ else } 0 \quad (9)$$

**The mean membership expectation** ( $\mathbb{E}_{MM}$ ) (Eq. 10) expect the presence of edges to be proportional to the lifetimes of the two nodes within the community. It is calculated as the geometric mean of the two lifetimes.

$$\mathbb{E}_{MM} [L_{uv \in C}] = \frac{k_u k_v}{2m} \frac{\sqrt{|T_{u \in C}| |T_{v \in C}|}}{|T|} \quad (10)$$

Given that  $|T_{uv \in C}| \leq \sqrt{|T_{u \in C}| |T_{v \in C}|} \leq |T_C|$ ,  $\mathbb{E}_{MM}$  can be interpreted as a compromise between  $\mathbb{E}_{CM}$ —that may promotes overly erratic temporal communities—and  $\mathbb{E}_{JM}$ —that promotes temporal communities with little or no evolution.

We can note that, in the special case where two nodes belong to the community in the same time,  $\mathbb{E}_{MM}$  is equivalent to  $\mathbb{E}_{CM}$ . Conversely, if one impose communities to stay unchanged during their time of existence, then  $\mathbb{E}_{MM} = \mathbb{E}_{CM} = \mathbb{E}_{JM}$ . Furthermore, if there is only one time step in the link stream, they are all equivalent to the expectation of the Modularity (Eq. 6), which is in line with our objective of generalization.

#### 4.4 Smoothness term

With no smoothness term for community discontinuities, modularity does not promote communities that are continuous over time (see Section 5). We argue that the purpose of considering temporal community structures is to capture information on their lifecycle, which necessitates communities that are continuous over time.

MS-Modularity smoothness term is based on the creation of interslice couplings between successive temporal instances of each node, then sums the weights

of these artificial edges that fall in-between communities, i.e., when a node changes affiliation between one snapshot and the next. As a consequence, the more nodes change community, the greater the term. Noting  $\omega_{urs}$  the interslice weight of a node  $u$  between snapshots  $r$  and  $s$ , MS-Modularity smoothness term is expressed as

$$\mathbb{S}_{\mathcal{C}} \propto \sum_{\mathcal{C} \in \mathcal{C}} \sum_{u \in V} \sum_{r, s \in \mathcal{S}^2} \omega_{urs} \delta_{us \in \mathcal{C}} \delta_{us \in \mathcal{C}} \quad (11)$$

By definition, in temporal networks,  $\omega_{urs} = 0$  if  $|r - s| > 1$ , i.e., when snapshots  $r$  and  $s$  are not successive.

In our approach, it is essential to address the continuity of communities without relying on artificial temporal edges between time steps. To achieve this, we propose calculating the **Community Switch Count** (CSC), denoted as  $\eta_u$ , for each node  $u$ . The CSC represents the number of times a node transitions out of a community and subsequently joins another community. More precisely,  $\eta_u$  is the number of communities visited by node  $u$ , minus one. This count also accounts for instances where a node revisits the same community multiple times. Fig. 2 illustrates examples of the CSC. In the critical case where nodes change communities at each interaction, the following inequality holds:  $0 \leq \eta_u \leq k_u - 1$  for each node  $u$ , and therefore  $0 \leq \sum_{u \in V} \eta_u \leq 2m - N$ .

To maintain generality, we propose measuring the time discontinuity of a dynamic community structure with

$$\rho(L, \mathcal{C}) = \frac{1}{2m} \sum_{u \in V} \eta_u(\mathcal{C}) \quad (12)$$

We propose using this measure as a smoothness term  $\mathbb{S}_{\mathcal{C}}$  for L-Modularity. In the scenario where there is only one time step in the link stream,  $\rho = 0$ , effectively generalizing Modularity. For the edge case mentioned above,  $\rho$  approaches 1. If there are even more discontinuities,  $\rho$  is not bounded. The smoothness term can be weighted according to specific requirements. A higher weight value results in greater penalization of discontinuities. For instance, with a weight of  $2m$  and only one node changing community, L-Modularity will be less than 0, thereby promoting stationary communities. Similarly, with a weight of 2, L-Modularity will be less than 0 if, on average, nodes change community every two interactions. The weighting of the smoothness term may be refined, and its impacts should be explored in future research.

#### 4.5 Extension to Multigraphs and Weighted graphs

Static Modularity naturally extends to multigraphs, by considering that the node degrees  $k_u$  correspond to the total number of edges, eventually repeated. Similarly, it naturally generalises to weighted graphs by using the node strengths (i.e., sum of weights of adjacent nodes) as values for  $k_u$ , and normalizing accordingly.

The same generalization can be done for L-Modularity. This generalization is particularly relevant in contexts in which temporal networks have been obtained

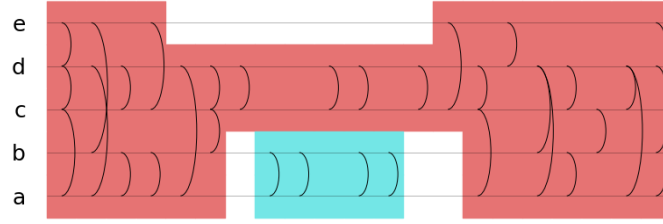


Fig. 2: **CSC Examples.** The figure is a 5 nodes link stream with 2 communities in red and blue. Nodes  $a$  and  $b$  leave the red community to form the two-nodes community in blue and then rejoin the red community again, so  $\eta_a = \eta_b = 2$ . Node  $e$  temporarily leaves the red community, so  $\eta_e = 1$ . Since  $c$  and  $d$  never change communities,  $\eta_c = \eta_d = 0$ . Finally,  $\rho = 0.08$ .

by aggregating observations over periods of time, e.g., counting social interactions between people over a day, a week, or a month. In this situation, one can represent the strengths of these observations using multigraphs or weights. Note that when generalizing to any type of weighted graphs, in particular with weights lower than 1, it might be necessary to renormalize the smoothness term.

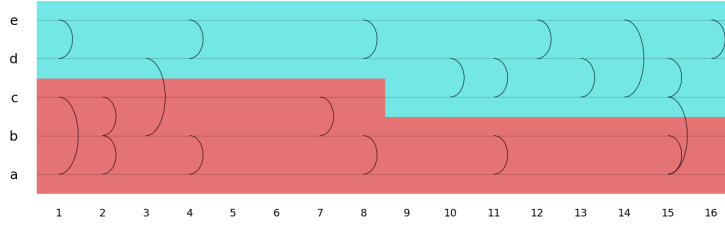
We will show in section 5.1 that, unlike MS-Modularity, L-Modularity provide coherent results in aggregated graphs compared with their non-aggregated versions (*Independence to time-aggregation property*).

## 5 Properties of dynamic communities

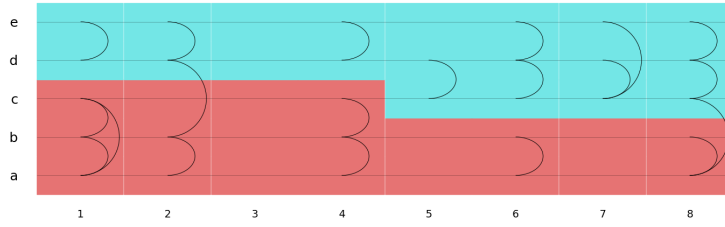
In the previous section, we proposed an adaptation of Modularity to link streams. In this section, we propose several properties that seem desirable for a definition of dynamic community structures in temporal networks and position L-Modularity relative to these properties.

Although the exact definition of what are *good* communities in static networks remains an open discussion, most authors agree on a general idea of groups *more strongly connected internally than the rest of the network*. Modularity is a mathematical transcription of this general principle —one among others [18] [21] [34][29]. One way to ensure that such a quantitative definition is compatible with the original intuition is to check that it respects some desired properties. For instance, one could say that Modularity in static networks respects the following properties:

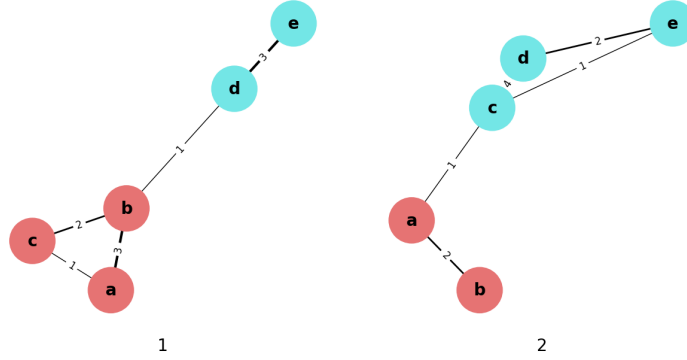
- The score does not favor *trivial* solutions, such as having all nodes in the same community or each node in its own community.
- If there are two separated connected components in the graph, Modularity necessarily attributes a lower score to the partition merging them than to another one identical in other aspects, but keeping them independent.



(a) Link stream, time granularity  $\Delta_t = 1$



(b) Link stream, time granularity  $\Delta_t = 2$



(c) Snapshots, time granularity  $\Delta_t = 8$

Fig. 3: Illustration of the independence of L-Modularity from the time granularity. All 3 examples are composed of the same interactions occurring at the same time, but aggregated at different time scales. The L-Modularity score is the same for the blue/red communities in all 3 cases, since 1)the number of interactions inside each community, 2)the relative duration of communities, 3)the global nodes degree, and 4)the number of affiliation discontinuity are all identical in all 3 cases.

- Communities are always denser than the overall network, as long as a solution is found with Modularity  $Q > 0$ .

We define similar properties for quality functions for link stream partitions.

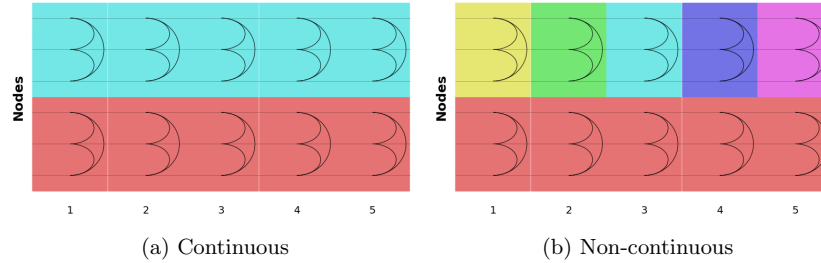


Fig. 4: Illustration of the *smoothness incentive* property. The same link stream is partitioned in different ways, each color representing a community. In Fig. (b), the three nodes on the top change communities, although there is not change in the network structure. A quality function respecting the smoothness incentive property must strictly favor partition (a) over (b)

### 5.1 Property 1: Independence to time-aggregation property

A common challenge in the study of temporal networks is defining a suitable time window for aggregation or determining an appropriate time granularity prior to performing dynamic community detection [22] [31] [19] [36] [10].

A notable feature of L-Modularity is that it yields the same score for a temporal node partition on different representations of the same link-stream obtained by different time-window aggregations without loss of information, as illustrated in Fig. 3. This property can be expressed as follows:

**Definition 3.** A dynamic community quality function is said to be *independent to time-aggregation* if, for two representations  $L$  and  $L'$ , such as  $L'$  is obtained by aggregating  $L$  into static graphs by time intervals —multiple interactions over a period being representing by weights or multi-edges— a partition defined on  $L'$  yields the same score on  $L$ .

A significant advantage of this property is that it makes L-Modularity a useful tool for comparing dynamic community structures on the same temporal network aggregated with different window sizes (see Section 6). In contrast, different MS-Modularity scores for community structures on the same temporal network with different aggregation window sizes are not comparable, as the multislice network changes each time.

### 5.2 Property 2: Smoothness Incentive

Addressing the smoothness of temporal communities is a well-known challenge in the field of dynamic community detection [13]. We propose that a quality function for temporal community structures must explicitly promote this smoothness, as illustrated in Fig. 4. More formally:

**Definition 4.** *Let's consider a temporal network  $L$  studied over a period  $T$ , split into two-time intervals  $T_1$  and  $T_2$ , and for which 1) the cumulated networks on both intervals are identical, 2) the partitions  $\mathcal{C}_\infty$  on  $T_1$  and  $\mathcal{C}_\infty$  on  $T_2$  in each interval are identical. A dynamic community quality function is said to have the **smoothness incentive** if, for a partition on  $T$ , it yields a strictly higher score to a partition in which each community  $C_i \in \mathcal{C}_\infty$  is merged with its corresponding community  $C_i \in \mathcal{C}_\infty$ , than to a partition in which any community in  $\mathcal{C}_\infty$  remains distinct from its equivalent one in  $\mathcal{C}_\infty$ .*

### 5.3 Property 3: Topochrone Disconnection Property

In static networks, non-connected components indicate that two subsets of nodes have no links between them. Modularity assigns a higher score when non-connected components belong to different communities. In temporal networks, *disconnected topochrone components* (Def. 5) occur when two sub-link streams do not share any nodes nor time intervals of existence. We propose that a quality function for temporal community structures should assign a higher score when disconnected topochrone components belong to different communities, rather than the same one.

**Definition 5.** *Two sub link streams  $L_1$  and  $L_2$  are said to be two **disconnected topochrone components** of a link stream if  $T_1 \cap T_2 = V_1 \cap V_2 = \emptyset$ .*

**Definition 6.** *A quality function for dynamic communities is said to respect the **topochrone disconnection** property if, given a partition in which a community contains two disconnected topochrone components, it gives a strictly higher score to the same partition in which those two topochrone components are split in two separate communities.*

For example, the three communities shown in figure 5a in blue, green, and purple are three disconnected topochrone components. A quality function respecting the topochrone disconnection property should strictly favor partition 5a over 5b.

### 5.4 L-Modularity and temporal community properties

As discussed in Section 4, L-Modularity results from combinations of the inclusion or non-inclusion of the smoothness term and the choice of longitudinal expectation. We will see which combinations respect the properties.

**L-Modularity and Independence to time-aggregation** L-Modularity is independent of time-aggregation, while MS-Modularity isn't.

This property naturally emerges from the elements used to compute L-Modularity, which depends only on the number of interactions inside communities, the time periods during which nodes belong to communities, and the total number of interactions per node.



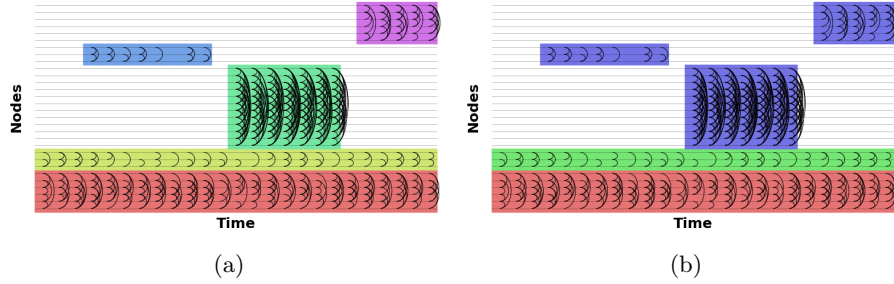


Fig. 5: **Topochrone disconnections.** A quality function respecting the Topochrone disconnection property should strictly favor the partition in Fig. a) over the one in b).

On the contrary, MS-Modularity depends 1) on the number of aggregation steps, since a higher number of snapshots leads to a higher number of interslice edges, encoding the smoothness term and 2) on the local degree of nodes at each step, and thus on the simultaneity of interactions, which is modified by aggregating.

It is important to note that while L-Modularity yields the same value for the same partition of the same link stream provided at multiple temporal granularities, the optimum partition might be different since a finer granularity allows to express partitions that would not be possible at coarser ones.

**L-Modularity and Smoothness Incentive** The smoothness incentive property is insured by the smoothness term  $\rho$  (Eq. 12), that penalizes unnecessary changes in affiliation.

**L-Modularity and Topochrone Disconnection** This property is insured by the expectation term of L-Modularity. Indeed, when using  $\mathbb{E}_{MM}$  (or  $\mathbb{E}_{JM}$ ), the number of edges expected inside a community grows with the time spent by nodes inside communities, even if this presence is not simultaneous. This corresponds to a vision of a perfect dynamic community as a set of nodes being homogeneously connected over a period. The topochrone disconnection goes against this objective. Note that for MS-Modularity, the situation relative to Topochrone disconnection is unclear and depends on modeling choices. If a node inactive at time  $t$  is removed completely from the network, then MS-Modularity does not fulfill the topochrone disconnection property, since no interslice edge will be added by merging the communities in a single one. If inactive nodes remain in the graph, then it respects this property, but will cause other difficulties, such as artificially rewarding the stability of communities without any observed internal edges.

### 5.5 Discussion on temporal communities properties

L-Modularity values for different term combinations have been calculated for all community structures shown in Figures 4 and 5. The results are summarized in table 1, which illustrates which properties are promoted by the three expectations and the presence or absence of the smoothness term. The results indicate the necessity of a smoothness term to promote smoothness incentives. Additionally, it is noted that the co-membership expectation (Eq. 8) does not promote robustness to topochrone disconnections. Therefore, we propose using either the joint membership expectation (Eq. 9) or the mean membership expectation (Eq. 10), along with the smoothness term (Eq. 12).

|                          | <b>Independance to time aggregation</b> | <b>Smoothness Incentive</b> | <b>Robust to topochrone disconnections</b> |
|--------------------------|---|-----------------------------|--|
| $\mathbb{E}_{CM}$        | ✓                                       | ✗                           | ✗  |
| $\mathbb{E}_{CM} + \rho$ | ✓                                       | ✓                           | ✗  |
| $\mathbb{E}_{JM}$        | ✓                                       | ✗                           | ✓  |
| $\mathbb{E}_{JM} + \rho$ | ✓                                       | ✓                           | ✓  |
| $\mathbb{E}_{MM}$        | ✓                                       | ✗                           | ✓  |
| $\mathbb{E}_{MM} + \rho$ | ✓                                       | ✓                           | ✓  |

Table 1: Properties promoted by the different terms introduced in Section 4: the expectations  $\mathbb{E}_{CM}$ ,  $\mathbb{E}_{JM}$ ,  $\mathbb{E}_{MM}$  and the smoothness term  $\rho$ .

## 6 Experimentation

In this section, we propose to evaluate L-Modularity using two different longitudinal expectations: the joint membership expectation (denoted as  $Q_{C,JM}$ , based on Eq. 9) and the mean membership (denoted as  $Q_{C,MM}$ , based on Eq. 10), both with a smoothness weight of 2. Specifically, we use the SocioPatterns high school dataset [24], which captures the contacts and friendship relations between students at a high school in Marseilles, France, during December 2013. The interactions are recorded as 20-second interval contacts among students from nine classes over five days. The dataset also includes information on the class membership of each student.

We apply two methods to uncover dynamic community structures within the dataset. The first method involves optimizing MS-Modularity, varying the interslice weight ratio. The second method employs a no-smoothing approach [13] (later, NS-Modularity) which first applies the Louvain algorithm [5] on each snapshot and then merges successive communities based on the Jaccard Index of their node sets. We experimented the NS-Modularity method varying the minimum threshold on the Jaccard Index for two communities to merge. Ultimately,

we compute the two versions of L-Modularity for each community structure obtained.

In order to use MS-Modularity and NS-Modularity, we need to perform network aggregation into snapshots. We use several steps, from days to 5-minute aggregation step. Due to computational difficulties, we were not able to perform some steps, and could not go lower than 5 minutes for MS-Modularity. Given that the data are based on interactions occurring exclusively during school hours over a span of five days, we included nighttime periods in our experiment. Each night begins after the final interaction of the day and ends before the first interaction of the following day. During these nighttime periods, no communities are present. Figures 6 and 7 illustrates the results.

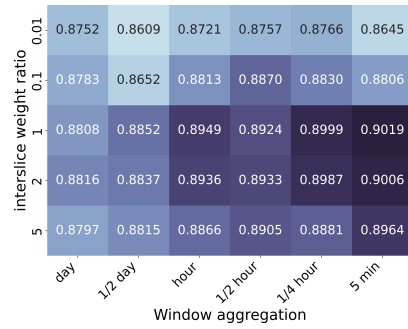
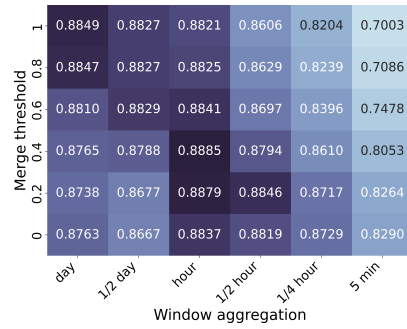
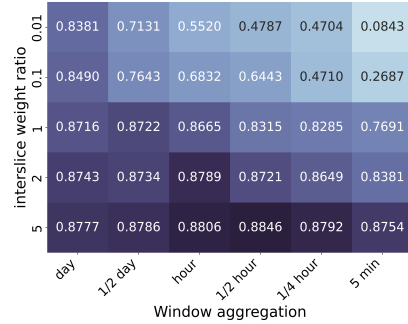
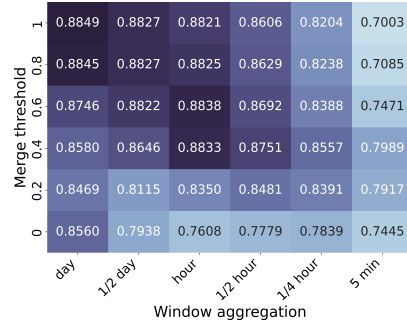
(a) MS method, evaluated by  $Q_{\mathcal{L},MM}$ (b) NS method, evaluated by  $Q_{\mathcal{L},MM}$ (c) MS method, evaluated by  $Q_{\mathcal{L},JM}$ (d) NS method, evaluated by  $Q_{\mathcal{L},JM}$ 

Fig. 6: Heatmaps of L-Modularity scores of dynamic community structures revealed by MS-Modularity optimization (MS) and No-smoothing method (NS) with varying parameters. Evaluations were conducted using two versions of L-Modularity: one,  $Q_{\mathcal{L},MM}$ , with the mean membership expectation and one,  $Q_{\mathcal{L},JM}$ , with the joint membership expectation.

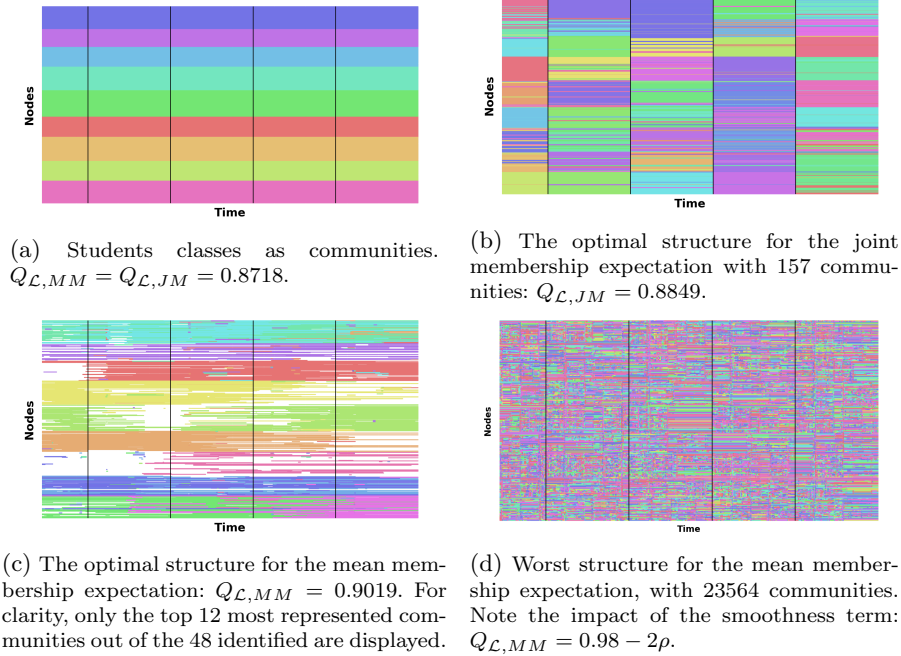


Fig. 7: Various dynamic community structures over the link stream of the Socipatterns High School dataset. Vertical black lines represent nights during which interactions, and therefore communities, cease to exist.

According to both L-Modularity versions, the MS-Modularity approach yields better community structures compared to the no-smoothing method. The advantage of MS-Modularity lies in its use of interslice weights, which facilitate the direct identification of continuous communities. In contrast, the no-smoothing method tends to overfit individual snapshots, and this overfitting is not adequately mitigated during the merging phase. This overfitting is exacerbated by more refined window aggregations. Figures 6a and 6c illustrate the challenge of selecting appropriate window sizes and interslice weights for the MS-Modularity approach. According to L-Modularity, the finest window sizes tends to capture the most detailed temporal dynamics, and reasonably high interslice weights favor community continuity. However, determining the optimal combination of these parameters is complex. For instance, the best community structure for  $Q_{\mathcal{C},MM}$  (Fig. 7b) was revealed with the finest window aggregation of 5 minutes and an interslice weight ratio of 1, which is not the highest interslice weight tested. In contrast, the best community structure for  $Q_{\mathcal{C},JM}$  was revealed with the highest interslice weight ratio of 5, and a window aggregation of 30 minutes, not the finest one. Indeed,  $Q_{\mathcal{C},JM}$  tends to favor almost stationary communities with little or no evolution over time, whereas  $Q_{\mathcal{C},MM}$  allows for more variations in dynamic community structures. Furthermore, the best community structure

for  $Q_{C, JM}$  was revealed with the NS method by aggregating data at the scale of a day and merging successive communities from each snapshot only if they share the exact same set of nodes (Fig. 7c). Both versions of L-Modularity can be used depending on specific analytical needs.

One could also consider the students' classes, provided in the dataset, as a *ground truth* partition (Fig. 7a). This partition yields a relatively high L-Modularity score, though not the highest. The experiment indicates that some students tend to interact beyond their class affiliations. Notably, Fig. 7a and Fig. 7c share the same smoothness value, as communities are considered to last the duration of each day, with only nights contributing to the smoothness term. However, Fig. 7c exhibits a higher L-Modularity score, demonstrating that at the daily scale, student communities do not strictly adhere to class boundaries.

Figure 7d illustrates the necessity of the smoothness term. It represents the best community structure according to L-Modularity when the smoothness term is discounted.

Note that we only compared L-Modularity scores of partitions obtained by methods that do not try to optimize it directly. The best partition discovered corresponds to something close to the expected ground truth given by classes, which shows that it does not fall into common traps such as favoring partitions overfitted at each timestep. However, one could expect to discover even better longitudinal communities, such as those distinguishing class periods from lunchtime and breaks between classes. This would require an algorithm able to better explore the space of possible partitions.

## 7 Discussion

We presented a quality function for dynamic community structures in link streams that does not require a predefined time scale for analysis or any preprocessing of the natural temporal interactions. This function is capable of handling various configurations of dynamic community structures. Furthermore, like Modularity, L-Modularity is directly generalizable to directed, weighted, multigraphs, and bipartite temporal networks.

We believe that the approach we presented for generalizing Modularity can be seen as a first step in the direction of defining quality functions for link streams and that similar work could be done for other quality functions such as the Map Equation [34]. As a generalization of Modularity, one might expect similar limitations, such as the resolution limit problem [15]. These limitations should be analyzed in future work.

As it is defined here, L-Modularity cannot be used directly for community detection, because no efficient method exists to explore the space of possible temporal partitions of a link stream. A natural next step would consist in developing a Louvain-like algorithm to explore the space of possible partitions, searching to maximize L-Modularity. We anticipate that the results will provide valuable feedback on the behavior of L-Modularity with both synthetic and real data.

Additionally, we hope this approach will uncover subtle dynamic community structures in real data, enhancing real-world data analysis.

## Acknowledgements

We would like to thank Matthieu Latapy for his valuable feedback and discussions. We also extend our gratitude to SAHAR for financing this project. Yasaman Asgari thanks the University of Zurich and the Digital Society Initiative for (partially) financing this project.

## Materials availability

The code necessary to reproduce the experiments and evaluate Longitudinal Modularity on various datasets is available here: [https://github.com/fondationsahar/dynamic\\_community\\_detection/](https://github.com/fondationsahar/dynamic_community_detection/).

## References

1. Arenas, A., Duch, J., Fernández, A., Gómez, S.: Size reduction of complex networks preserving modularity. *New Journal of Physics* 9(6), 176–176 (Jun 2007), <http://dx.doi.org/10.1088/1367-2630/9/6/176>
2. Aynaud, T., Guillaume, J.L.: Multi-step community detection and hierarchical time segmentation in evolving networks. In: *Knowledge Discovery and Data Mining* (2011), <https://api.semanticscholar.org/CorpusID:15958971>
3. Barber, M.J.: Modularity and community detection in bipartite networks. *Physical Review E* 76(6) (Dec 2007), <http://dx.doi.org/10.1103/PhysRevE.76.066102>
4. Béres, F., Pálovics, R., Oláh, A., Benczúr, A.A.: Temporal walk based centrality metric for graph streams. *Applied network science* 3, 1–26 (2018)
5. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008(10), P10008 (Oct 2008), <http://dx.doi.org/10.1088/1742-5468/2008/10/P10008>
6. Boudebza, S., Cazabet, R., Azouaou, F., Nouali, O.: Olcpm: An online framework for detecting overlapping communities in dynamic social networks. *Computer Communications* 123, 36–51 (2018)
7. Boudebza, S., Cazabet, R., Nouali, O., Azouaou, F.: Detecting stable communities in link streams at multiple temporal scales. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. pp. 353–367. Springer (2019)
8. Bovet, A., Delvenne, J.C., Lambiotte, R.: Flow stability for dynamic community detection. *Science advances* 8(19), eabj3063 (2022)
9. Brissette, C., Pandey, U., Slota, G.M.: Effects of null model choice on modularity maximization. In: *International Conference on Complex Networks and Their Applications*. pp. 261–272. Springer (2023)
10. Caceres, R., Berger-Wolf, T., Grossman, R.: Temporal scale of processes in dynamic networks. pp. 925–932 (12 2011)

11. Cazabet, R., Amblard, F., Hanachi, C.: Detection of overlapping communities in dynamical social networks. In: 2010 IEEE second international conference on social computing. pp. 309–314. IEEE (2010)
12. Cazabet, R., Borgnat, P., Jensen, P.: Enhancing space-aware community detection using degree constrained spatial null model. In: Complex Networks VIII: Proceedings of the 8th Conference on Complex Networks CompleNet 2017 8. pp. 47–55. Springer (2017)
13. Cazabet, R., Boudebza, S., Rossetti, G.: Evaluating community detection algorithms for progressively evolving graphs. *Journal of Complex Networks* 8(6), cnaa027 (2020)
14. Delvenne, J.C., Yaliraki, S.N., Barahona, M.: Stability of graph communities across time scales (2009)
15. Fortunato, S., Barthélemy, M.: Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104(1), 36–41 (Jan 2007), <http://dx.doi.org/10.1073/pnas.0605965104>
16. Fortunato, S., Hric, D.: Community detection in networks: A user guide. *Physics reports* 659, 1–44 (2016)
17. Gauvin, L., Génois, M., Karsai, M., Kivelä, M., Takaguchi, T., Valdano, E., Vestergaard, C.L.: Randomized reference models for temporal networks. *SIAM Review* 64(4), 763–830 (Nov 2022), <http://dx.doi.org/10.1137/19M1242252>
18. Holland, P.W., Laskey, K.B., Leinhardt, S.: Stochastic blockmodels: First steps. *Social Networks* 5(2), 109–137 (1983), <https://www.sciencedirect.com/science/article/pii/0378873383900217>
19. Krings, G., Karsai, M., Bernhardsson, S., Blondel, V.D., Saramäki, J.: Effects of time window size and placement on the structure of an aggregated communication network. *EPJ Data Science* 1(1) (May 2012), <http://dx.doi.org/10.1140/epjds4>
20. Latapy, M., Viard, T., Magnien, C.: Stream graphs and link streams for the modeling of interactions over time. *Social Network Analysis and Mining* 8, 1–29 (2018)
21. von Luxburg, U.: A tutorial on spectral clustering (2007), <https://arxiv.org/abs/0711.0189>
22. Léo, Y., Crespelle, C., Fleury, E.: Non-altering time scales for aggregation of dynamic networks into series of graphs (05 2018)
23. Ma, X., Dong, D.: Evolutionary nonnegative matrix factorization algorithms for community detection in dynamic networks. *IEEE transactions on knowledge and data engineering* 29(5), 1045–1058 (2017)
24. Mastrandrea, R., Fournet, J., Barrat, A.: Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLOS ONE* 10(9), e0136497 (Sep 2015), <http://dx.doi.org/10.1371/journal.pone.0136497>
25. Matias, C., Rebafka, T., Villers, F.: A semiparametric extension of the stochastic block model for longitudinal networks. *Biometrika* 105(3), 665–680 (2018)
26. Mucha, P.J., Richardson, T., Macon, K., Porter, M.A., Onnela, J.P.: Community structure in time-dependent, multiscale, and multiplex networks. *Science* 328(5980), 876–878 (May 2010), <http://dx.doi.org/10.1126/science.1184819>
27. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. *Physical review E* 69(2), 026113 (2004)
28. Paul, S., Chen, Y.: Null models and community detection in multi-layer networks (2020), <https://arxiv.org/abs/1608.00623>
29. Peixoto, T.P.: Descriptive vs. inferential community detection in networks: pitfalls, myths and half-truths. Cambridge University Press (2023)

30. Poda, V., Matias, C.: Comparison of modularity-based approaches for nodes clustering in hypergraphs. *Peer Community Journal* 4 (2024)
31. Ribeiro, B., Perra, N., Baronchelli, A.: Quantifying the effect of temporal resolution on time-varying networks. *Scientific reports* 3, 3006 (10 2013)
32. Rossetti, G., Cazabet, R.: Community discovery in dynamic networks: a survey. *ACM computing surveys (CSUR)* 51(2), 1–37 (2018)
33. Rossetti, G., Pappalardo, L., Pedreschi, D., Giannotti, F.: Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning* 106, 1213–1241 (2017)
34. Rosvall, M., Axelsson, D., Bergstrom, C.T.: The map equation. *The European Physical Journal Special Topics* 178(1), 13–23 (Nov 2009), <http://dx.doi.org/10.1140/epjst/e2010-01179-1>
35. Rosvall, M., Axelsson, D., Bergstrom, C.T.: The map equation. *The European Physical Journal Special Topics* 178(1), 13–23 (2009)
36. Scholtes, I., Wider, N., Garas, A.: Higher-order aggregate networks in the analysis of temporal networks: path structures and centralities. *The European Physical Journal B* 89(3) (Mar 2016), <http://dx.doi.org/10.1140/epjb/e2016-60663-0>
37. Viard, T., Latapy, M., Magnien, C.: Computing maximal cliques in link streams. *Theoretical Computer Science* 609, 245–252 (2016)
38. Yang, T., Chi, Y., Zhu, S., Gong, Y., Jin, R.: Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine learning* 82, 157–189 (2011)