

System Identification for Temporal Networks

Shvydun, Sergey; Miegheem, Piet Van

DOI

[10.1109/TNSE.2023.3333007](https://doi.org/10.1109/TNSE.2023.3333007)

Publication date

2023

Document Version

Final published version

Published in

IEEE Transactions on Network Science and Engineering

Citation (APA)

Shvydun, S., & Miegheem, P. V. (2023). System Identification for Temporal Networks. *IEEE Transactions on Network Science and Engineering*, 11(2), 1885-1895. <https://doi.org/10.1109/TNSE.2023.3333007>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

<https://www.openaccess.nl/en/you-share-we-take-care>

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

System Identification for Temporal Networks

Sergey Shvydun  and Piet Van Mieghem , *Fellow, IEEE*

Abstract—Modelling temporal networks is an open problem that has attracted researchers from a diverse range of fields. Currently, the existing modelling solutions of time-evolving graphs do not allow us to provide an accurate graph sequence. In this paper, we examine the network dynamics from a system identification perspective. We prove that any periodic graph sequence can be accurately modelled as a linear process. We propose two algorithms, called Subspace Graph Generator (SG-gen) and Linear Periodic Graph Generator (LPG-gen), for modelling periodic graph sequences and provide their performance on artificial graph sequences. We further propose a novel model, called Linear Graph Generator (LG-gen), that can be applied to non-periodic graph sequences. Our experiments on artificial and real networks demonstrate that many temporal networks can be accurately approximated by periodic graph sequences.

Index Terms—Network dynamics, system identification, temporal networks.

I. INTRODUCTION

MANY real systems such as social, financial, biological and technological systems can be represented as networks, where the nodes are the elements of the system and the links are interactions between them. The power of networks resides in their ability to provide insights into complex system structure and to model complex dynamics such as diffusion and contagion. However, most studies in graph theory are performed under the assumption that the structure of the network is static, which is not true for most real systems. Indeed, in many applications, the systems are dynamic in nature and evolve over time which, in turn, affects their topology and the processes that propagate over the network. Such an observation leads to the fact that many practical problems can be solved more accurately if the time-evolving structure of the graph is taken into account. The ability to model real systems facilitates our understanding of the nature and the timing of observed evolution, but it may also provide some useful intuition about the future behaviour of the system, thereby making valuable predictions. Moreover, understanding the graph evolution can identify system malfunction or security intrusion.

Manuscript received 26 July 2023; revised 12 October 2023; accepted 13 November 2023. Date of publication 15 November 2023; date of current version 23 February 2024. This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under Grant 101019718. Recommended for acceptance by Dr. W. Du. (*Corresponding author: Sergey Shvydun.*)

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: S.Shvydun@tudelft.nl; P.F.A.VanMieghem@tudelft.nl).

This article has supplementary downloadable material available at <https://doi.org/10.1109/TNSE.2023.3333007>, provided by the authors.

Digital Object Identifier 10.1109/TNSE.2023.3333007

Understanding the evolution of networks is still an open problem. Most papers on network dynamics focus on temporal link prediction using matrix factorization, probabilistic approaches, spectral clustering, time series or deep learning methods that fail to capture global topological features and non-linear varying temporal patterns of the network [1]. Other approaches focus on the graph level and consists in producing generative graph sequences that mimic the real-world networks in terms of certain topological features such as the number of links, clustering coefficient, degree distribution, connected components or motifs [2], [3], [4], [5], [6], [7], [8], [9]. Various approaches have been proposed to model particular dynamical processes such as human mobility [10], [11], [12], [13] or communication networks [14]. These models are mostly probabilistic and activity-driven (nodes and links are active or inactive within some time intervals) or spatiotemporal (nodes are changing their position in space with respect to some trajectory) [15], [16]. For instance, Chang et al. [11] introduce a Markov Modulated Process (MMP), where states of the Markov chain encode certain modifications to the original graph. It is shown that an MMP model captures global graph properties, but it does not provide an accurate topology of the graph and cannot be applied for modelling processes that have seasonality or spikes around certain events. To the best of our knowledge, none of the existing models generates graph sequences that resemble the real graph in terms of its set of adjacency matrices. Moreover, most models do not provide knowledge about the underlying process.

This paper is aimed as one step forward towards a deeper understanding of network evolutionary processes. We examine network dynamics from a system identification perspective. Based on the observed graph sequence, we attempt to model the graph evolution as a linear process. The goal of the study is to find a process that generates such graph sequences accurately.

Many real world systems possess a dynamics that repeats during a certain period of time due to a daily rhythm. Examples of such quasi-periodic systems are road traffic, computer networks, logistics, human mobility, social interactions and many others. Ma and Hellerstein [17] show that periodic patterns often lead to actionable insights about the evolutionary process. We show that any periodic graph evolution can be described by a linear time-invariant process.

Our major contributions can be summarized:

- We model the graph dynamics as a *linear process* using a system identification approach in Section III-B.
- We prove in Section III-D that any periodic graph sequence can be modelled exactly. We provide information about the dimensionality of the system that produces the exact graph dynamics.

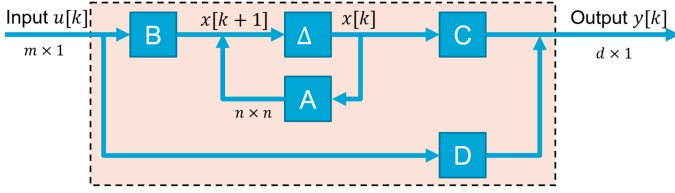


Fig. 1. Visualization of an LTI model. The symbol Δ represents a delay.

- We propose SG-gen, LPG-gen and LG-gen algorithms to model periodic and non-periodic graph sequences (see Sections III-B, III-E, and III-F).
- We test SG-gen, LPG-gen and LG-gen algorithms on various artificial and real graph sequences and illustrate their reachable accuracy (Sections III-G–III-H).

The paper is organized as follows. In Section II, we provide basic information about time-invariant state-space model and subspace methods for system identification. In Section III, we propose the SG-gen model for graph generation and apply it to artificial graph dynamics. Next, we consider the properties of the SG-gen model and provide the exact solution for periodic graph sequences (LG-gen). We also consider the application of LG-gen to quasi-periodic artificial networks and real graphs. Finally, Section IV concludes. Additionally, we introduce the notation to the reader in the Appendix A.

II. LINEAR TIME-INVARIANT STATE-SPACE MODEL

A. Problem Statement

The dynamics of a linear system in discrete time k is defined by a linear time-invariant (LTI) state-space model [18]:

$$\begin{cases} x[k+1] = A \cdot x[k] + B \cdot u[k], \\ y[k] = C \cdot x[k] + D \cdot u[k], \end{cases} \quad (1)$$

where $u[k] \in \mathbb{R}^m$ is the input vector, $y[k] \in \mathbb{R}^d$ is the output vector, $x[k] \in \mathbb{R}^n$ is the state vector and where $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{d \times n}$ and $D \in \mathbb{R}^{d \times m}$ are time-invariant matrices that define the relation between input, output and state vectors. The dimension n of the state vector $x[k]$ defines the order of an LTI system. Equivalently, (1) is

$$\begin{bmatrix} x[k+1] \\ y[k] \end{bmatrix} = Q \cdot \begin{bmatrix} x[k] \\ u[k] \end{bmatrix}, \quad \text{with } Q = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \quad (2)$$

The block matrix Q is an $(d+n) \times (n+m)$ system matrix. The scheme of an LTI system from [19] is represented in Fig. 1.

In general, as shown by Verhaegen and Verdult [18], there are different state representations that yield the same dynamic relation between observations $u[k]$ and $y[k]$. In fact, any system

$$\begin{cases} x_p[k+1] = A_p \cdot x_p[k] + B_p \cdot u[k], \\ y[k] = C_p \cdot x_p[k] + D_p \cdot u[k], \end{cases}$$

with $A_p = P^{-1}AP$, $B_p = P^{-1}B$, $C_p = CP$, $D_p = D$ and $x_p[k] = P^{-1}x[k]$ for any non-singular similarity matrix P is an equivalent system to (1) since it produces the same output

$y[k]$ given the same input $u[k]$ and merely relabels their vector components. Therefore, it is only possible to identify the matrices A, B, C, D up to a similarity (or state) transformation P and any LTI system with system matrices (A_p, B_p, C_p, D_p) and state vector x_p is equivalent to (1).

The identification of an LTI state-space model in (1) from input and output measurements can be solved using subspace identification methods [20]. The subspace methods are based on the fact that, by storing the input and output data in structured block Hankel matrices, it enables to retrieve certain vector subspaces that are related to the system matrix Q of an LTI model. More precisely, the input and output measurements can be stored in structured block Hankel matrices as follows:

$$Y_{i,s,h} = \begin{bmatrix} y[i] & y[i+1] & \cdots & y[i+h-1] \\ y[i+1] & y[i+2] & \cdots & y[i+h] \\ \vdots & \vdots & \ddots & \vdots \\ y[i+s-1] & y[i+s] & \cdots & y[i+h+s-2] \end{bmatrix},$$

where $Y_{i,s,h} \in \mathbb{R}^{sd \times h}$, h and s are parameters of an LTI model (in general, $s \ll h$). The blocked Hankel matrix $U_{i,s,h} \in \mathbb{R}^{sm \times h}$ constructed from $u[t]$ is defined in a similar way. Block Hankel matrices allow us to rewrite (1) as

$$Y_{i,s,h} = \Gamma_s \cdot X_{i,h} + H_s \cdot U_{i,s,h}, \quad (3)$$

where $X_{i,h} = \begin{bmatrix} x[i] & x[i+1] & \cdots & x[i+h-1] \end{bmatrix} \in \mathbb{R}^{n \times h}$, $\Gamma_s = \begin{bmatrix} C & CA & \cdots & CA^{s-1} \end{bmatrix}^T \in \mathbb{R}^{sd \times n}$ and $H_s \in \mathbb{R}^{sd \times sm}$ are correspondingly the extended observability and the block Toeplitz matrices derived from A, B, C, D :

$$H_s = \begin{bmatrix} D & 0 & \cdots & 0 \\ CB & D & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{s-2}B & CA^{s-1}B & \cdots & D \end{bmatrix}.$$

Several solutions of (3) have been presented in [19], [20], [21], [22], [23], [24]. These methods express the row space of $Y_{i,s,h}$ as a linear combination of row spaces of $X_{i,h}$ and $U_{i,s,h}$. Subspace methods benefit from reliable numerical algorithms such as LQ decomposition and the singular value decomposition (SVD), which are non-iterative and do not need nonlinear optimization techniques. One of the most prevailing algorithms for subspace system identification is a numerical algorithm for subspace state space system identification (N4SID) [24], which is described in Appendix B. The N4SID algorithm can be extended to identify an LTI system corrupted by process and measurement noise [18].

III. APPLICATION OF THE LTI MODEL TO TEMPORAL NETWORKS

A. Preliminaries

We consider a temporal graph, denoted by $G_k(\mathcal{N}, \mathcal{L}_k)$, consisting of a set \mathcal{N} of N nodes (vertices) connected by a set \mathcal{L}_k of L_k links (edges) at discrete time k . The graph G_k is described

by an $N \times N$ adjacency matrix A_k whose elements $a_{ij}(k)$ are either one or zero depending on whether there is a link between nodes i and j or not. For simplicity, we assume that the set of nodes is fixed and the graph is undirected, then all adjacency matrices $A_k = A_k^T$ are real symmetric matrices. Additionally, we denote by $\mathcal{L} = \bigcup_{k=1}^T \mathcal{L}_k$ the union of all L links that emerged in graphs over T time slots with $L \leq \binom{N}{2}$. The problem lies in identifying the underlying process that generates the sequence G_1, G_2, \dots, G_T of graphs over T time slots.

B. Subspace Model for Temporal Networks

We assume that the graph dynamic is described by an LTI state-space model. The $L \times 1$ binary vector $a[k]$ specifies¹ the links in graph G_k at discrete time k with component $a_i[k] = 1$ if the i -th link of the set \mathcal{L} is present in graph G_k , otherwise $a_i[k] = 0$. Then we can construct $T - 1$ input-output sequence $\{(u[k], y[k])\}_{k=1}^{T-1}$, where $u[k] = a[k]$ and $y[k] = a[k + 1]$. In other words, the input of the LTI model at discrete time k is the structure of graph G_k and the output is the structure of graph G_{k+1} . The N4SID algorithm estimates the system matrix Q that best approximates the graph G_{k+1} given the real graph G_k .

Here, we propose a novel model, called *Subspace Graph Generator (SG-gen)*, that generates the whole graph sequence using the identified LTI model. The SG-gen model assumes that the estimated output vector $\hat{y}[k]$ at discrete time k equals the input vector $\hat{u}[k + 1]$ at discrete time $k + 1$, i.e., $\hat{y}[k] = u[k + 1]$. The governing (2) simplifies to

$$\begin{bmatrix} x[k + 1] \\ \hat{y}[k] \end{bmatrix} = \begin{bmatrix} x[k + 1] \\ u[k + 1] \end{bmatrix} = Q \cdot \begin{bmatrix} x[k] \\ u[k] \end{bmatrix}, \quad (4)$$

where Q is an $(n + L) \times (n + L)$ system matrix. If we denote the block vector $v[k] = \begin{bmatrix} x[k] \\ u[k] \end{bmatrix}$, then the law (4) of SG-gen is

$$v[k + 1] = Q \cdot v[k], \quad (5)$$

whose solution follows by iteration as

$$v[k + 1] = Q^k \cdot v[1]. \quad (6)$$

The output sequence $\hat{y}[1], \dots, \hat{y}[T - 1]$ corresponding to the graphs $\hat{G}_2, \dots, \hat{G}_T$ can be computed via (6). The SG-gen model is visualized in Fig. 2.

The rationale behind the SG-gen model is that if the LTI model accurately identifies the structure of graph G_{k+1} from graph G_k , it can also generate the whole graph sequence G_2, \dots, G_T using the initial structure of the graph G_1 .

The SG-gen model has some limitations. SG-gen is not identical to a classical LTI system, which only makes a 1-step prediction and does not use the estimated output $\hat{y}[k]$ as the input for the next time slot $k + 1$. On the contrary, SG-gen relates to an autonomous system, does not require any external inputs after the 1st time slot and may produce infinite graph sequences.

¹The vector $a[k]$ does not preserve information about the position of links in G_k , except if we choose its dimension equal to $\binom{N}{2}$, in which case a component $a_i[k]$ corresponds to an element $(A_k)_{pr}$ in the adjacency matrix.

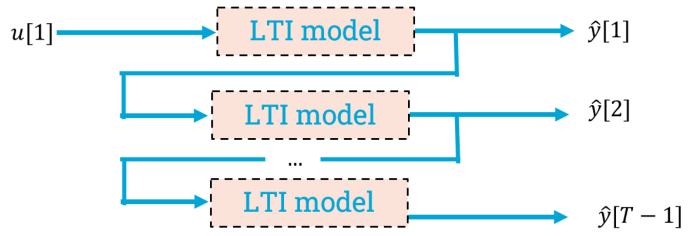


Fig. 2. Output generation using SG-gen model.

Finally, since SG-gen is based on the N4SID algorithm, it requires input parameters (the size of the block Hankel matrix) and should satisfy all assumptions of N4SID.

Property of the SG-gen linear system. If the $q \times q$ system matrix Q with $q = n + L$ has q distinct eigenvalues $\lambda_1, \dots, \lambda_q$, then the matrix Q can be written [25] as

$$Q = X\Lambda Y^T,$$

where the matrices X and Y contain the right- and left-eigenvectors of Q , respectively, in their columns and obey $XY^T = I$ and $\Lambda = \text{diag}(\lambda_i)$ is a diagonal matrix of eigenvalues of Q . Since $Q^k = X\Lambda^k Y^T$ and applied to (6), it follows that

- $v[k] \rightarrow 0$ for $k \rightarrow \infty$ iff $|\lambda_i| < 1, \forall i = 1, \dots, q$.
- $v[k] \rightarrow \infty$ for $k \rightarrow \infty$ if $\exists i$ such that $|\lambda_i| > 1$.

Hence, this property of SG-gen constitutes the main limitation, because not all graph dynamics can be generated by SG-gen. Indeed, if there exists at least one eigenvalue larger than 1, then the long-term evolution of SG-gen will tend to infinity and the SG-gen model becomes useless. If all eigenvalues are less than 1, the SG-gen model will eventually generate empty graphs, which are not suitable for real-world networks. Therefore, SG-gen imposes strict requirements on the graph dynamics, namely $|\lambda_i| = 1$ for all $1 \leq i \leq q$, and only periodic graph sequences can be modelled by SG-gen (see Section III-D).

Finally, we assess the performance of SG-gen by the *mean square error (MSE)*

$$MSE(y, \hat{y}) = \frac{1}{T-1} \sum_{k=1}^{T-1} \sum_{i=1}^L (y_i[k] - \hat{y}_i[k])^2, \quad (7)$$

where $y[k]$ and $\hat{y}[k]$ are the real and estimated vectors corresponding to the graph G_{k+1} at time $k + 1$. Thus, the accuracy of SG-gen is estimated based on all output measurements.

C. Experiments on Periodic Graph Sequences

We apply the SG-gen model to various periodic graph sequences. We assume that the temporal graph has a period p ; thus, $G_k = G_{k+p}$ for $\forall k = 1, \dots, T - p$. We begin with periodic sequences, where the graph G_{k+1} is constructed from graph G_k by adding or removing only one link. Next, we consider more comprehensive periodic sequences, where we make multiple random changes or none in graph G_k to generate G_{k+1} .

- Graph dynamic 1:* we arbitrary enumerate all possible links $L = N(N - 1)/2$ in the graph and add 1 link per time slot to the graph in the prescribed order. Once the graph G_k equals the complete graph K_N , precisely

one link is removed per time slot in the reversed order. Once the graph is empty, we repeat the graph generation process. The described process has a period $p = 2L$. SG-gen has been tested on networks containing up to 25 nodes.

- *Graph dynamic 2*: the dynamical process is the same as for the graph dynamic 1, but we also allow the graph to keep its structure unchanged for some time slots. For instance, we assume that the graph does not change for 6 time slots if it is empty, for 4 time slots if it is complete and for 2 time slots if it contains half the number of possible links. The period of the process is $p = 2L + 12$. SG-gen has been tested on networks containing up to 10 nodes.
- *Graph dynamic 3*: we start with an arbitrary graph G_1 , then produce up to 1 random change in a graph structure per time slot but after p time slots, it becomes identical to G_1 . The period of the process is $p = 2L$. The SG-gen model has been applied to graphs with up to 11 nodes while the MSE is averaged across 500 iterations.

A detailed information about the experiments on periodic data is provided in the Appendix C. Overall, SG-gen almost exactly predicts the graphs sequence G_2, \dots, G_T from graph G_1 for all graph dynamics, i.e., $MSE(y, \hat{y}) \approx 0$. Moreover, we observe that the choice of the initial graph G_1 does not affect the results of the experiment. The experiments indicate that the minimal number of periods to identify the system matrix Q with accurate predictions does not exceed 3 for the graph dynamic 1, 9 for the graph dynamic 2 and 7 for the graph dynamic 3. The state vectors $x[1], \dots, x[T-1]$ has the same period p .

Next, we examine how the dimension n of the state vector $x[k]$ and the order of the system matrix Q depend upon the number L of links in a temporal graph. For the graph dynamic 1, the state vector has L coordinates and Q is a $p \times p$ matrix. For the graph dynamic 2, the dimension of the state vector is $n = p - L - 1$ and Q is a $(p - 1) \times (p - 1)$ matrix. For the graph dynamic 3, the dimension n satisfies $|p - L - n| \leq 1$ while the order of Q is either p or $p - 1$.

The experiments in Appendix C show that SG-gen generates many periodic graph sequences accurately. However, we have not yet proved that SG-gen can model *all* periodic graph sequences, because the SG-gen model has limitations. First, the identification of the system matrix Q is constructed based on a 1-step prediction from the real input. Thus, there is no guarantee that the identified matrix Q will accurately generate G_2, \dots, G_T sequence from G_1 . Second, since the SG-gen model is based on the N4SID algorithm, the performance of SG-gen depends on the choice of input parameters such as the number of observations in the dataset and the number of rows in block Hankel matrices. In general, there is no prior information about these parameters, consequently, we determine them experimentally by finding the values that minimize the MSE in (7). In Section III-D, we prove that all periodic graph sequences can be modelled by SG-gen and we propose the LPG-gen algorithm that identifies accurately the system matrix Q without subspace methods.

D. Properties of a Linear System for Periodic Data

First, we analyse the $(n + L) \times (n + L)$ system matrix Q . Since the vector state $x[k]$ has a period p , it holds that $v[k + p] = v[k]$, for $\forall k = 1, \dots, T - p$. Combined with $v[k + p] = Q^p \cdot v[k]$ in (6), the matrix Q satisfies $Q^p = I$, where I is the identity matrix. The eigenvalue equation $Q \cdot z = \lambda z$, where λ is an eigenvalue of Q belonging to eigenvector z , is equivalent to $Q^p \cdot z = \lambda^p z$ for non-negative integer p and leads, with $Q^p = I$, to $\lambda^p - 1 = 0$, whose solution is $\lambda = e^{2\pi i m/p}$ where $m = 0, 1, \dots, p - 1$. In other words, all $p = n + L$ eigenvalues of the matrix Q are unique and evenly spaced around the unit circle in the complex plane with $\lambda = 1$ as real eigenvalue corresponding to $m = 0$. Only if p is even, then $\lambda = -1$ is the other possible real eigenvalue for $m = p/2$.

The $(k + 1)$ -th output of SG-gen is given by $v[k + 1] = Q^k v[1]$ in (6), where $v[1]$ is a concatenation of vectors $x[1]$ and $u[1]$. In general, vector $u[1]$ is an $L \times 1$ vector that corresponds to the graph G_1 , while the $n \times 1$ vector $x[1]$ is unknown. Thus, we need to determine the dimension n as well as the values of the vector $x[1]$.

Lemma 1: If the $(n + L) \times 1$ vectors sequence $v[1], \dots, v[T]$ has a period p , then $v[k + 1] = Q^k v[1]$ in (6) implies that

- $Q^p = I$. The matrix Q is called $(p + 1)$ -potent [26];
- All vectors $v[k]$ with $k = 1, \dots, T$ are eigenvectors of matrix $Q^p = I$, because $v[k + p] = Q^p \cdot v[k] = v[k]$.

Since the vectors $v[1], \dots, v[T]$ are periodic, it is sufficient to consider only first p vectors during 1 period. We rewrite (5) as

$$\begin{cases} v[k + 1] = Q \cdot v[k], & \forall k = 1, \dots, p - 1, \\ v[p + 1] = Q \cdot v[p] = v[1], \end{cases}$$

or, equivalently

$$[v[2] \dots v[p] \ v[1]] = Q \cdot [v[1] \dots v[p-1] \ v[p]]. \quad (8)$$

We denote the two $(n + L) \times p$ matrices $V_1 = [v[1] \dots v[p-1] \ v[p]]$ and $V_2 = [v[2] \dots v[p] \ v[1]]$ such that (8) becomes $V_2 = Q \cdot V_1$.

Lemma 2: Let $v[1], \dots, v[p]$ be a linearly independent set of $p \times 1$ vectors or, equivalently, the matrices V_1 and V_2 are of full rank. Then, there exists a unique matrix Q satisfying (8), which is defined by

$$Q = V_2 \cdot V_1^{-1}. \quad (9)$$

Relation (9) is crucial for periodic graph sequences. Indeed, if vectors $v[1], \dots, v[p]$ are linearly independent in p -dimensional vector space, any periodic graph sequence can be accurately modelled by the system matrix Q . This result explains our previous findings in Section III-C, where the order of the system matrix Q is exactly p and the dimension n of the state vector $x[k]$ is $p - L$. Thus, p is a sufficient dimension for the SG-gen model to exactly determine the output of the periodic sequence. In fact, the identification of matrix Q can be performed by constructing the $n \times 1$ vectors $x[1], \dots, x[p]$ such that the vectors $v[1], \dots, v[p]$ will be linearly independent. The system matrix Q can be computed using (9). Consequently, the N4SID algorithm is not required for periodic sequences. In general, it is not evident

that p is a minimal order of Q . Experiments show that some graph dynamics can be accurately generated by mapping vectors $v[1], \dots, v[p]$ into an r -dimensional vector space with $r < p$. Theorem 3 determines the minimal order of matrix Q .

We denote by $C(V)$ the $(n + L) \times p$ circulant matrix of the vectors $V = \{v[1], \dots, v[p]\}$,

$$C(V) = \begin{bmatrix} v[1] & v[2] & \cdots & v[p-1] & v[p] \\ v[2] & v[3] & \cdots & v[p] & v[1] \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ v[p] & v[1] & \cdots & v[p-2] & v[p-1] \end{bmatrix}.$$

Theorem 3: Let $V = \{v[1], \dots, v[p]\}$ be a set of $r \times 1$ vectors with $r \leq p$. If there exists a non-zero matrix Q that satisfies (8), then the minimal order r of the system matrix Q is defined as

$$r = \text{rank}(C(U)),$$

where $C(U)$ is a $(p \cdot L) \times p$ circulant matrix of input vectors $U = \{u[1], \dots, u[p]\}$ corresponding to the graphs G_1, \dots, G_p . The $r \times 1$ vectors $v[1], \dots, v[p]$ must satisfy

$$C(V) \cdot \begin{bmatrix} \alpha_1 \\ \dots \\ \alpha_p \end{bmatrix} = 0. \quad (10)$$

Theorem 3 implies that $v[1], \dots, v[p]$ can always be mapped into an r -dimensional vector space with $r \leq p$. The rows of the matrix $C(U)$ contains all the dynamics of links within p time slots, hence, we call the rank of $C(U)$ the *complexity* of the periodic process. By introducing vectors $x[1], \dots, x[p]$, satisfying (10), we can uniquely identify the system matrix Q .

Theorem 4: Let $V = \{v[1], \dots, v[p]\}$ be a set of $r \times 1$ vectors that satisfies (10) and $r \leq p$. Then, there exists a unique matrix Q , satisfying (8),

$$Q = [v[k_1 + 1] \dots v[k_r + 1]] \cdot [v[k_1] \dots v[k_r]]^{-1}, \quad (11)$$

where $v[k_1], \dots, v[k_r]$ are r linearly independent vectors.

Theorems 3 and 4 are proved in Appendix D.

E. Linear Periodic Graph Generator (LPG-Gen)

Theorems 3 and 4 are the basis for a general algorithm that identifies the system matrix Q for periodic graph sequences. Moreover, the rows of the matrix $C(U)$ from Theorem 3, which are linearly independent from the first row space of matrix $[u[1], \dots, u[p]]$, can be used as state vectors. Algorithm 1 proposes the *Linear Periodic Graph Generator* (LPG-gen) that produces accurately any periodic graph sequence:

Similarly to the SG-gen model, LPG-gen is designed only for periodic graph sequences. However, LPG-gen ensures that any periodic graph sequences can be modelled accurately and the order r of the matrix Q is minimal. Additionally, LPG-gen does not require input parameters.

The computational complexity of LPG-gen. Step 1 requires at most $\frac{L \cdot p \cdot T}{2}$ operations. Steps 2-3 takes at most $2Lp^3$ operations [27]. Finally, the worst-case running time of step 4 is $2p^3$. Therefore, the estimated computational complexity of LPG-gen is $O(LpT + Lp^3)$. Experimentally, LPG-gen has a lower runtime than SG-gen (see the Appendix E).

Algorithm 1: Linear Periodic Graph Generator (LPG-Gen).

Input: vectors $u[1], \dots, u[T]$ corresponding to G_1, \dots, G_T .

Output: system matrix Q and initial state vector $x[1]$.

1. Define the minimal period p such that $u[k] = u[k+p]$, for $\forall k = 1, \dots, T - p$.
2. Solve $\alpha_1, \dots, \alpha_p$ in the system

$$C(U) \cdot \begin{bmatrix} \alpha_1 \\ \dots \\ \alpha_p \end{bmatrix} = 0. \quad (12)$$

The order r of the matrix Q equals the rank $C(U)$. Define the pivot variables $\alpha_{k_1}, \dots, \alpha_{k_r}$ from (12).

3. Initialize the state vectors $x[k_1], \dots, x[k_r]$ to make vectors $v[k_1], \dots, v[k_r]$ linearly independent. The remaining $p - r$ vectors $x[k]$ with $k = 1, \dots, p$ and $k \notin \{k_1, \dots, k_r\}$ are initialized with respect to (10).
4. Compute the system matrix Q via (11).

Return: $Q, x[1]$.

F. Linear Graph Generator (LG-gen)

In previous Sections, we have discussed only perfect periodic graph sequences that obey $G_k = G_{k+p}$ for any discrete time $k = 1, \dots, T - p$ and period $p < T$. Most real-world systems are not perfectly periodic, hence, the graphs G_k and G_{k+p} are not exactly the same. Therefore, we extend the LPG-gen model to non-periodic graph sequences.

Since LPG-gen accurately produces any periodic graph sequence, it serves as a basis for its extension. We borrow a powerful idea from Fourier analysis that has found application in many disciplines: most signals can be decomposed into periodic waveforms. Zygmund [28, Chapter 17] treats Fourier series in the m -dimensional Euclidean space and states that the extension of a single variable $m = 1$ to the case of several variables $m > 1$ is 'generally' the same.

We apply the *periodicity transform* (PT) that decomposes signals into basic periodic components by projecting onto a set of periodic subspaces [29]. More precisely, the periodicity transform iteratively defines the closest p -periodic vector to the initial vector and then applies the PT to the residuals. Contrary to the Fourier transform, periodic subspaces are not orthogonal, hence, the periodicity transform does not in general provide a unique representation. However, Sethares and Staley [29] show that, in many cases, the periodicity transform provides a clearer explanation of the underlying nature of the signals than the Fourier transform.

The *Linear Graph Generator* (LG-gen) approximates the real graph sequence G_1, \dots, G_T by a set of periodic graph sequences. We denote by $G_1^{(i)}, \dots, G_T^{(i)}$ an i -th periodic graph sequence satisfying $G_k^{(i)} = G_{k+p_i}^{(i)}$ for any $k = 1, \dots, T - p_i$ with period p_i . Then, we approximate a graph G_k at discrete time k by l periodic graph sequences:

$$G_k \approx \sum_{i=1}^l G_k^{(i)}. \quad (13)$$

The rationale behind the LG-gen model is that many non-periodic graph evolutions may contain periodic patterns

Algorithm 2: Linear Graph Generator (LG-gen).

Input: vectors $u[1], \dots, u[T]$ corresponding to G_1, \dots, G_T , number of periodic graph sequences l .

Output: system matrices Q_1, \dots, Q_l and initial state vectors $x_1[1], \dots, x_l[1]$.

1. $u \leftarrow [u[1], \dots, u[T]]$
2. for $i \leftarrow 1$ to l
 - a) Determine the period p_i for u
 $p_i \leftarrow \text{LocalMin}(u)$.
 - b) Construct the average sequence with period p_i
 $\bar{u} \leftarrow [\bar{u}[1], \dots, \bar{u}[T]]$.
 - c) Identify the system that produces \bar{u} :
 $Q_i, x_i[1] \leftarrow \text{LPG-gen}(\bar{u})$.
 - d) Update $u \leftarrow u - \bar{u}$

Return: Q_1, \dots, Q_l and $x_1[1], \dots, x_l[1]$.

that describe certain dynamical processes in the network. The observed dynamics with periodic patterns can be approximated by (13), while each periodic sequence $G_1^{(i)}, \dots, G_T^{(i)}$ for $1 \leq k \leq T$ is computed by LPG-gen.

The LG-gen model is proposed in Algorithm 2. First, LG-gen identifies the optimal period length p_1 and periodic graph sequence $G_1^{(1)}, \dots, G_T^{(1)}$ that best approximates the initial graph sequence G_1, \dots, G_T . The obtained periodic graph sequence is computed by LPG-gen. Next, we construct the residual graph sequence $G_1 - G_1^{(1)}, \dots, G_T - G_T^{(1)}$ which corresponds to information about graph structure that is not captured by the periodic graph sequence $G_1^{(1)}, \dots, G_T^{(1)}$. Next, the same procedure is iteratively applied to estimate the residual sequence. The algorithm continues until l periodic graph sequences are obtained. If the residual sequence $G_1 - \sum_i G_1^{(i)}, \dots, G_T - \sum_i G_T^{(i)}$ is an empty graph sequence, then $G_k = \sum_{i=1}^l G_k^{(i)}$ is exact for a finite l . Algorithm 2 can be also applied to weighted and directed networks.

An important part of LG-gen is the identification of periodic patterns in temporal networks. Andres et al. [30] identify periodic time scales by computing the Fourier transform of the function that measures the portrait divergence between successive temporal networks. Here, we apply the Algorithm 3, called *LocalMin*, that iteratively defines the optimal period p^* based on the vectors $u[1], \dots, u[T]$. First, for each possible period $p = 1, \dots, T$, *LocalMin* constructs the average periodic pattern $\bar{u} = (\bar{u}[1], \dots, \bar{u}[p])$ by

$$\bar{u}[k] = \frac{1}{n_k} \sum_{i=0}^{n_k-1} u[k + i \cdot p], \quad (14)$$

where $n_k = \lfloor (T - k)/p \rfloor + 1$ is the total number of vectors $u[1], \dots, u[T]$ that correspond to the k -th element of the periodic graph sequence \bar{u} . Here $\lfloor s \rfloor$ is the largest integer smaller than or equal to s . The general idea of the average periodic sequence construction is presented in Fig. 3. Second, *LocalMin* replicates average patterns \bar{u} until T time slots and computes the MSE between $u[1], \dots, u[T]$ and each replicated periodic sequence

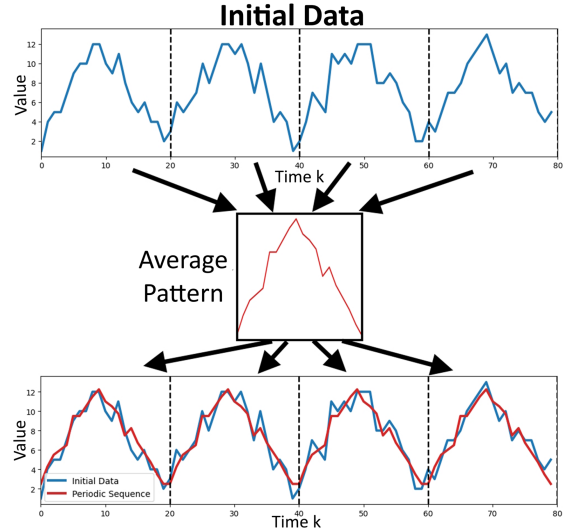


Fig. 3. Identification of period for non-periodic data.

$\bar{u}[1], \dots, \bar{u}[T]$. Finally, the optimal period p^* is given by

$$p^* = \arg \max_{p \leq T/2} \frac{\text{count}[p]}{\lfloor (T-1)/p \rfloor + 1},$$

where $\text{count}[p]$ computes the total number of timestamps $k \in \{p, 2p, \dots, (\lfloor (T-1)/p \rfloor + 1) \cdot p\}$ corresponding to the local minima of the MSE.

The computational complexity of LG-gen is $O(l \cdot L \cdot T^2 + L \cdot \sum_{i=1}^l p_i^3)$ where $l \cdot L \cdot T^2$ operations are needed by *LocalMin*. An important feature of *LocalMin* is its convergence. Theorem 5 guarantees the decrease in the residual values of the remaining graph sequence $G_1 - \sum_i G_1^{(i)}, \dots, G_T - \sum_i G_T^{(i)}$. We also prove that the periodic sequence based on the average provides the largest decrease of the MSE. Sethares and Staley [29] have shown that the average periodic pattern is an orthogonal projection of the input vector onto the p -periodic subspace.

Theorem 5: Let $u = \{u[1], \dots, u[T]\}$ be a set of T vectors with dimension L corresponding to G_1, \dots, G_T . For any non-zero average sequence $\bar{u} = \{\bar{u}[1], \dots, \bar{u}[T]\}$ with an arbitrary period $p \leq T$, it holds that

- a) $\|u - \bar{u}\| \leq \|u\|$,
- b) $\|u - \bar{u}\| < \|u - \tilde{u}\|$.

where $\tilde{u} = \{\tilde{u}[1], \dots, \tilde{u}[T]\}$ is any sequence of period p such that $\bar{u} \neq \tilde{u}$.

The proof of Theorems 5 is deferred to Appendix F. Additionally, Appendix F.3 examines the minimal period p to obtain a non-zero average periodic sequence and discusses the convergence of LG-gen for arbitrary vector sequences.

G. Experiments on Non-Periodic Graph Sequences

We consider two graph dynamics that are almost periodic. For simplicity, both dynamics are based on the periodic graph

Algorithm 3: Identification of Periodic Patterns (LocalMin).**Input:** vectors $u[1], \dots, u[T]$ corresponding to G_1, \dots, G_T **Output:** period p^* $loss \leftarrow 0_{T \times 1}$ for $p \leftarrow 1$ to T

a) Construct the average periodic sequence by (14)

$$\bar{u} \leftarrow [\bar{u}[1], \dots, \bar{u}[T]].$$

b) $loss[p] \leftarrow mse(u, \bar{u})$ $count \leftarrow 0_{T/2 \times 1}$ for $p \leftarrow T/2$ $j \leftarrow p$ while $j < T$ if $loss[j] < loss[j-1]$ and $loss[j] < loss[j+1]$ $count[p] \leftarrow count[p] + 1$ $j \leftarrow j + p$

$$count[p] \leftarrow \frac{count[p]}{[(T-1)/p]+1}$$

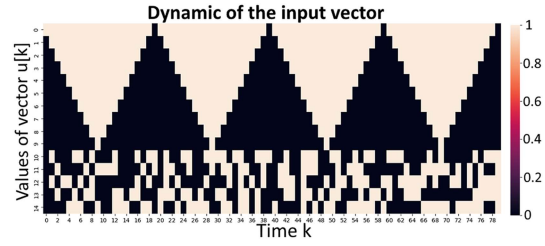
 $p^* \leftarrow \arg \max_p (count[p])$ **Return:** p^* .

Fig. 4. Graph Dynamic 4 (6 nodes). Among 15 possible links only 10 links have periodic patterns while 5 other links evolve randomly.

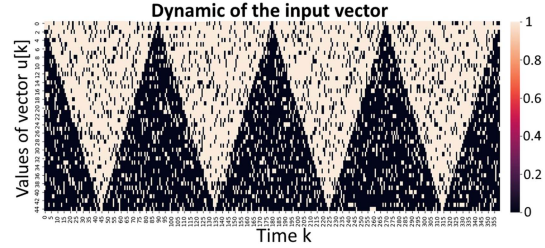


Fig. 5. Graph Dynamic 5 (10 nodes). 8 random changes are performed at each time slot.

dynamic 1, but the choice of the periodic dynamic does not influence the results of the section.

- *Graph dynamic 4:* we assume that q links evolve periodically with respect to the graph dynamic 1, while the remaining links in the graph G_k evolve randomly at each time slot k (see Fig. 4).
- *Graph dynamic 5:* we consider the graph dynamic 1 and then perform q random changes in the graph structure G_k at each time slot k (see Fig. 5).

The graph dynamic 4 indicates how the performance of the SG-gen and LG-gen models is affected by the presence of random links. The SG-gen model has been tested on graphs with up to 20 nodes. SG-gen can produce an exact dynamic for q periodic links if the number of random links is low. For a graph dynamic from Fig. 4 the only difference occurs for the last 5 links ($MSE \approx 0.93$). However, the increase of the number of random links results in the inability to identify periodic behaviour for q links by SG-gen.

Fig. 6 illustrates the results of LG-gen on the graph from Fig. 4 for $l = 1$ and $l = 3$. LG-gen for $l = 1$ is comparable to SG-gen. However, if the dynamic is approximated by 3 periodic graph sequences, LG-gen provides a much better performance ($MSE \approx 0.59$). Overall, the increase of parameter l leads to a more accurate graph identification (e.g.: $MSE \approx 0.2$ for $l = 8$). The first periodic graph sequence captures the periodic behaviour of q links while other periodic graph sequences model the dynamics for the last 5 links. Finally, LG-gen still captures periodic patterns for q links even if 85% of links evolve at random.

The SG-gen model was applied to the graph dynamic 5 and did not provide an accurate performance ($MSE \approx 3.92$). SG-gen provides an empty graph after 7 time slots.

Fig. 7 illustrates the results of LG-gen on the graph from Fig. 5 for $l=1$ and $l=5$. Similarly to the graph dynamic 4, the first periodic graph sequence of LG-gen approximates the

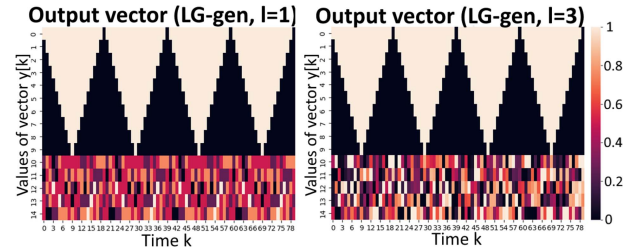


Fig. 6. Output of the LG-gen model for the Graph Dynamic 4 (6 nodes).

original almost periodic dynamic while additional periodic graph sequences are used to model the remaining changes. The MSE equals 2.08 for $l = 1$, 1.02 for $l = 5$ and 0.43 for $l = 10$. Moreover, if we apply the rounding to the sum of generated sequences, 7 periodic graph sequences are sufficient to provide an ideal performance for unweighted graph, i.e., $G_k = \text{round}(\sum_{i=1}^7 G_k^{(i)})$ for any $k = 1, \dots, T$, where $\text{round}(\cdot)$ denotes for rounding to the nearest integer.

H. Experiments on Real Data

We now examine the performance of LG-gen to real networks. We consider several face-to-face interaction networks, collected in a school, in a hospital and in the workspace by the SocioPatterns project (<http://www.sociopatterns.org/>). The data were gathered using wearable RFID badges, which assess the proximity of two individuals with a probability in excess of 99% over an interval of 20 seconds [31]. We show that real networks can be accurately modelled by the LG-gen model.

1. *A school in Lyon:* The LyonSchool dataset [32] contains the contact events between 242 individuals (232 children and 10 teachers) in a primary school in Lyon, France, during two days

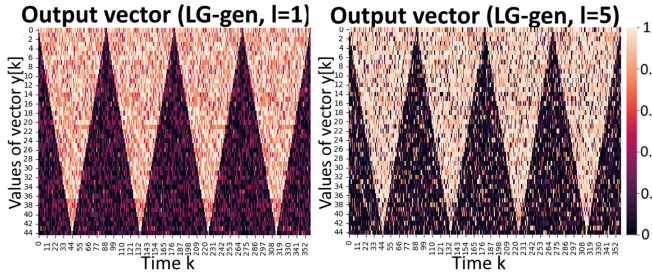


Fig. 7. Output of the LG-gen model for the Graph Dynamic 5 (10 nodes).

TABLE I
MSE FOR LYONSCHOOL NETWORK

Model	Number of periodic graphs sequences l								
	1	2	3	4	5	6	7	8	9
No rounding	26.5	20.6	16.4	15.1	13.1	11.8	11.2	10.9	10.7
With rounding	37	16.7	6.8	3.4	1.07	0.3	0.1	0.04	0.01

TABLE II
MSE FOR HOSPITAL NETWORK

Model	Number of periodic graphs sequences l								
	1	2	3	4	5	6	7	8	9
No rounding	0.89	0.87	0.78	0.68	0.6	0.55	0.51	0.47	0.43
With rounding	0.95	0.91	0.28	0.06	0.03	0.01	0.005	0.002	0.001

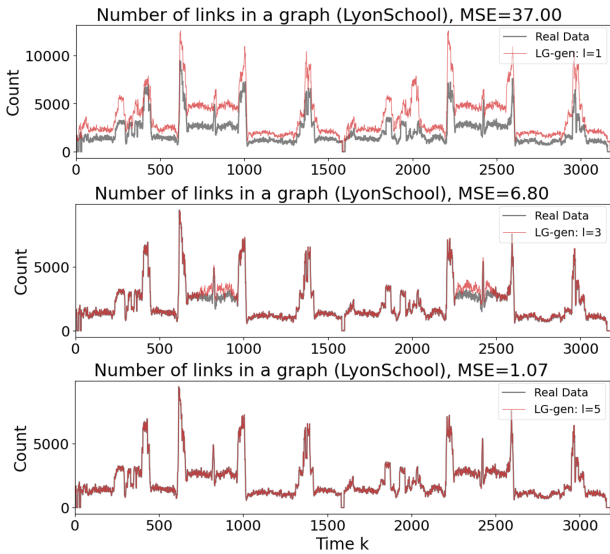


Fig. 8. Performance of LG-gen (with rounding) on LyonSchool.

in October 2009. Each time slot of the network corresponds to a 20-second interval $[t - 20 s, t]$ while the connections between nodes represent active contacts during the interval. All the contacts have occurred between 10:30 and 19:20. Thus, total number of time slots is 3,180. The total number of links is 6,594,492 with approximately 2,110 connections per time slot. The total number of unique links in the graph is 26,594. The average number of changes between two adjacent graphs G_k and G_{k+1} is 1,245.

Intuitively, since the contact data is based on a two-days period, we assume that the period should not exceed 1 d. Thus, we test LG-gen with 1,590 time slots (≈ 8 hours 50 minutes) as an upper bound of the period. The initial MSE between LyonSchool and a zero graph sequence is 43.7.

The results of the LG-gen model are presented in Table I. Additionally, Fig. 8 illustrated the performance of LG-gen in

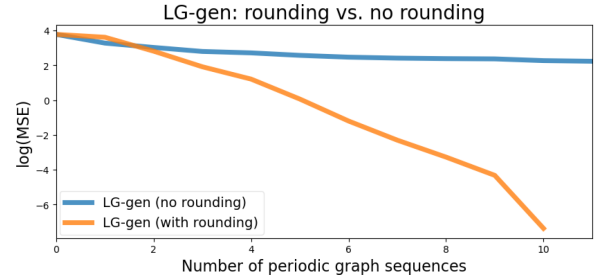


Fig. 9. Logarithm of the MSE on LyonSchool network.

terms of the dynamics of the link count and the MSE, which is computed between original and generated graph sequences via (7). First, the LG-gen model captures around 30% of links if it approximates the real dynamics using only 1 periodic graph sequence. Second, we observe that the rounding provides a lower MSE for any $l \geq 2$. Fig. 9 illustrates that the MSE decreases exponentially if the rounding is applied at the final stage of LG-gen. For $l = 5$, the difference between LG-gen and real network is about 1.15 links in average. The period of 4 graph sequences is in the range from 1,488 to 1,590 time slots (≈ 1 d cycle), the remaining period is 819 time slots ($\approx 1/2$ d cycle).

Finally, the LG-gen model generates the graph sequence accurately for $l = 11$, i.e., $G_k = \text{round}(\sum_{i=1}^{11} G_k^{(i)})$, for any $k = 1, \dots, T$. Thus, the LyonSchool network can be accurately modelled by the LG-gen model.

2. Hospital Data: The dataset [33] contains information about contacts between patients, patients and health-care workers (HCWs) and among HCWs in a hospital ward in Lyon, France over 4 days (from December 6, 2010 at 1:00 pm to December 10, 2010 at 2:00 pm). The total number of nodes is 75. Each time slot corresponds to a 20-second interval $[t - 20 s, t]$ while connections between nodes represent active contacts during different time intervals. The total number of time slots is 17,396 (9,453 time slots with contacts). The total number of contacts is 32,424. There are 4 peaks in the data that correspond to daily cycles. Therefore, we can assume that the dynamics can be approximated by 1 d periods. The initial MSE between Hospital network and a zero graph sequence is 0.95.

Table II demonstrates the performance of LG-gen. The increase of l decrease the MSE and for $l \geq 5$ we obtain almost exact matching. In general, the rounding at the final step of LG-gen produces a lower MSE for all $l \neq 1$. Even for $l = 9$ the generated graph sequence is not exact, which can be explained by the fact that contacts in hospital have less periodic patterns in general, thus, requiring more periodic sequences to capture

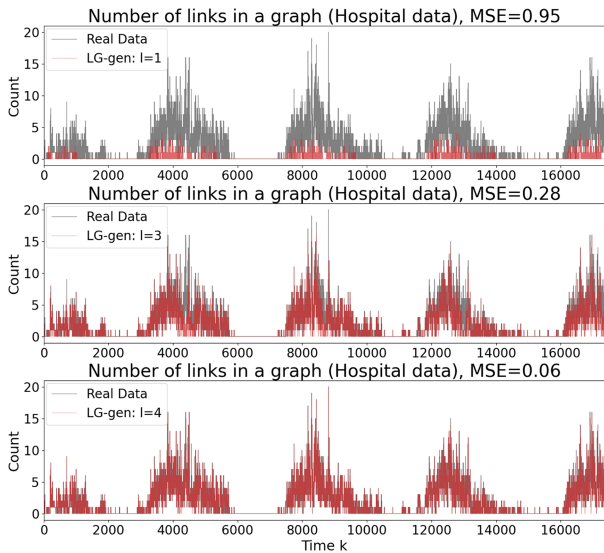


Fig. 10. Performance of LG-gen (with rounding) on Hospital network.

TABLE III
MSE FOR INVS13 NETWORK

Model	Number of periodic graphs sequences l							
	1	2	3	4	5	6	7	8
No rounding	0.018	0.014	0.012	0.01	0.008	0.007	0.00064	0.0058
With rounding	0.024	0.012	0.007	0.002	0.001	0.0005	0.0004	0

such dynamics. Finally, the period of all approximated graph sequences varies from 3,950 to 4,320 time slots (≈ 1 d cycle).

Fig. 10 illustrates information about the number of links in generated graph (after rounding) and in the real data. We observe that for $l = 5$ LG-gen shows a good correspondence with the real Hospital network.

3. *InVS13 dataset*: The dataset contains the human contact events between 95 individuals in the office building in France in 2013, namely a building of the *Institut de veille sanitaire* [34]. The total number of time slots is 51,120 (20,129 time slots have at least one contact) that correspond to a 2 week period. We consider 10 working days from 9.00 a.m. to 9.00 p.m (23,220 time slots in total). Additionally, we have excluded the first 100 minutes from our experiment, which looked anomalous, as there have been observed around 264 contacts per time slot for the first 300 time slots and only 15.9 contacts per time slot for the remaining time slots. In average, the number of changes between graphs G_k and G_{k+1} is 10.7. We observe a weekly periodicity in the data, thus, we have tested the LG-gen model with 10,805 time slots (5 working days) as an upper bound of the maximum period. The MSE between the workspace network and a zero graph sequence is approximately 0.0245.

The performance of the LG-gen model for $l \leq 8$ is provided in Table III. Again, we observe that the initial graph can be well approximated by a set of periodic graph sequences. The rounding at the final step provides a lower MSE for all l excluding $l = 1$. For $l = 5$ the MSE is 0.001 which corresponds to approximately 525 link difference in total during 22,920 time

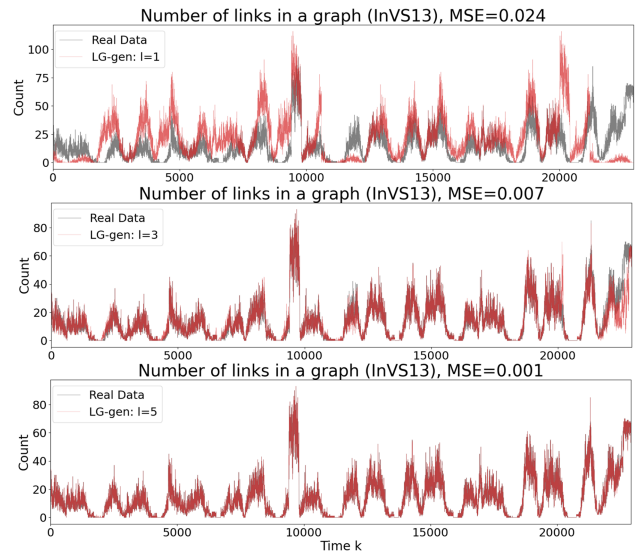


Fig. 11. Performance of LG-gen (with rounding) on InVS13 network.

slots. For $l = 8$ the generated graph sequence is exact if the rounding step is applied. The period of all approximated graph sequences varies from 7,219 (≈ 3.3 days cycle) to 10,796 time slots (≈ 5 days cycle). Finally, Fig. 11 demonstrates the difference in the number of changes between original data and LG-gen with rounding.

Overall, the LyonSchool, Hospital and InVS13 networks can be accurately approximated by a small number of periodic graph sequences.

IV. DISCUSSION

We have modelled the dynamics of temporal networks as a *linear process* and have proposed 3 algorithms² to generate various graph sequences G_1, \dots, G_T accurately:

- 1) *Subspace Graph Generator (SG-gen)* for periodic data, which defines the matrix Q by the N4SID algorithm;
- 2) *Linear Periodic Graph Generator (LPG-gen)* for periodic data, which defines the matrix Q via (11);
- 3) *Linear Graph Generator (LG-gen)* for non-periodic data, which approximates graph G_k at discrete time k by l periodic graph sequences while each periodic sequence is computed by LPG-gen.

We started with periodic graph sequences and have proved that *any* p -periodic dynamics can be accurately reproduced by SG-gen and LPG-gen. Both algorithms are designed only for periodic dynamics and generate exactly the same graph sequences while LPG-gen has a lower runtime and does not require any additional input parameters. The order of the system matrix Q is defined by the rank of the circulant matrix $C(U)$ of input vectors $U = \{u[1], \dots, u[p]\}$. Additionally, we propose the rank of $C(U)$ as a measure of the *complexity* of the periodic process.

²The Python code is available at <https://github.com/SergSHV/>

The LG-gen model has been tested on artificial and real networks. The results of our experiments on artificial data demonstrate a good performance of LG-gen. Since many real systems have periodic patterns, we observe that most of them can be well approximated by the relatively small number of periodic graph sequences.

The strength of SG-gen, LPG-gen and LG-gen is their simplicity as they are linear. An important issue of LG-gen is the minimal number l of periodic graph sequences that are used for the approximation. We observe that the rounding at the final step may dramatically (about exponentially!) decrease the required value l . In fact, if all residuals are in the interval $(-0.5; 0.5)$, LG-gen does not require any additional steps. Finally, all the algorithms can be applied to directed and weighted networks.

There are some limitations of the LG-gen model. A first concern is the assumption that the graph dynamic is almost periodic. If the process is stable and does not contain any cyclic events, the performance of LG-gen will decrease. However, the presence of periodic behaviour in many real world networks (such as human mobility, social interactions, traffic networks, etc.) makes the LG-gen model quite useful. Another drawback of LG-gen is its high computational complexity. Identification of periodic cycles in the dataset is not a trivial problem, the current implementation requires $l \cdot L \cdot T^2$ operations where L is the number of links in a graph, T is the total number of time slots and l is the number of periodic graph sequences. Periodicity identification is one of the future steps of the research. Moreover, the order of the matrix Q depends on the complexity of the observed dynamics and is limited by the period p , which can be a very large number in general. For instance, if time slots corresponds to minutes, the maximal size of the matrix Q in the case of the weekly periodicity will be around $10^4 \times 10^4$. One way to solve this problem is to decrease the period length of the graph sequence (e.g.: daily cycles instead of weekly cycles) or to decrease the number of time slots (e.g.: time interval is 60 seconds instead of 20 seconds). Finally, SG-gen, LPG-gen and LG-gen assume that the nodes in the network remain the same in each time slot. In real systems, however, both nodes and links are added or removed over time. To remediate this problem, we can consider nodes as isolated during discrete times of inactivity.

We would like to point out that our research presents a conceptual framework showing that periodic graph sequences can be modelled as a linear process while non-periodic graph sequences can be accurately represented as a linear combination of periodic sequences.

ACKNOWLEDGMENT

The authors are grateful to M. Verhaegen, G. Leus and S. Dahlgren for their useful comments and suggestions.

REFERENCES

- [1] A. Divakaran and A. Mohan, "Temporal link prediction: A survey," *New Gener. Comput.*, vol. 38, pp. 213–258, 2020.
- [2] F. Bois and G. Gayraud, "Probabilistic generation of random networks taking into account information on motifs occurrence," *J. Comput. Biol.*, vol. 22, no. 1, pp. 25–36, 2015.
- [3] S. Purohit, L. B. Holder, and G. Chin, "Temporal graph generation based on a distribution of temporal motifs," in *Proc. 14th Int. Workshop Mining Learn. Graphs*, 2018, pp. 1–7. [Online]. Available: <https://www.mlworkshop.org/2018/>
- [4] A. Bojchevski, O. Shchur, D. Zügner, and S. Günnemann, "NetGAN: Generating graphs via random walks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 610–619.
- [5] D. Zhou, L. Zheng, J. Han, and J. He, "A data-driven graph generative model for temporal interaction networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 401–411.
- [6] G. Zeno, T. La Fond, and J. Neville, "DYMOND: Dynamic motif-nodes network generative model," in *Proc. Web Conf.*, 2021, pp. 718–729.
- [7] Y. Du et al., "GraphGT: Machine learning datasets for graph generation and transformation," in *Proc. Conf. Neural Inf. Process. Syst. Datasets Benchmarks*, 2021, pp. 1–29.
- [8] A. Longa, G. Cencetti, S. Lehmann, A. Passerini, and B. Lepri, "Neighbourhood matching creates realistic surrogate temporal networks," 2022, *arXiv:2205.08820*.
- [9] P. Liu and A. Sariyüce, "Using motif transitions for temporal graph generation," in *Proc. 29th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2023, pp. 1501–1511.
- [10] H. Barbosa et al., "Human mobility: Models and applications," *Phys. Rep.*, vol. 734, pp. 1–74, 2018.
- [11] B. Chang et al., "Markov modulated process to model human mobility," in *Proc. Int. Conf. Complex Netw. Appl.*, 2022, pp. 607–618.
- [12] A. Panisson, A. Barrat, C. Cattuto, W. Van den Broeck, G. Ruffo, and R. Schifanella, "On the dynamics of human proximity for data diffusion in ad-hoc networks," *Ad Hoc Netw.*, vol. 10, no. 8, pp. 1532–1543, 2012.
- [13] G. Mauro, M. Luca, A. Longa, B. Lepri, and L. Pappalardo, "Generating mobility networks with generative adversarial networks," *EPJ Data Sci.*, vol. 11, 2022, Art. no. 58.
- [14] G. Laurent, J. Saramäki, and M. Karsai, "From calls to communities: A model for time-varying social networks," *Eur. Phys. J. B*, vol. 88, 2015, Art. no. 301.
- [15] P. Holme, "Modern temporal network theory: A colloquium," *Eur. Phys. J. B*, vol. 88, pp. 1–30, 2001.
- [16] D. Ghosh et al., "The synchronized dynamics of time-varying networks," *Phys. Rep.*, vol. 949, pp. 1–63, 2022.
- [17] S. Ma and J. L. Hellerstein, "Mining partially periodic event patterns with unknown periods," in *Proc. IEEE Int. Conf. Data Eng.*, 2001, pp. 205–214.
- [18] M. Verhaegen and V. Verdult, *Filtering and System Identification: A Least Squares Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2007.
- [19] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems: Theory, Implementation, Applications*. Dordrecht, The Netherlands: Kluwer Academic, 1996.
- [20] B. De Moor, J. Vandewalle, M. Moonen, L. Vandenberghe, and P. Van Miegheem, "A geometrical strategy for the identification of state space models of linear multivariable systems with singular value decomposition," in *Proc. 8th IFAC/IFORS Symp. Identification Syst. Parameter Estimation*, 1988, pp. 493–497.
- [21] M. Moonen, B. De Moor, L. Vandenberghe, and J. Vandewalle, "On and offline identification of linear state space models," *Int. J. Control*, vol. 49, no. 1, pp. 219–232, 1989.
- [22] W. Larimore, "Canonical variate analysis in identification, filtering, and adaptive control," in *Proc. IEEE 29th Conf. Decis. Control*, 1990, pp. 596–604.
- [23] M. Verhaegen and P. Dewilde, "Subspace model identification Part 1. The output-error state-space model identification class of algorithms," *Int. J. Control*, vol. 56, pp. 1187–1210, 1992.
- [24] P. Van Overschee and B. De Moor, "'N4SID" subspace algorithms for the identification of combined deterministic stochastic system," *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.
- [25] P. Van Miegheem, *Graph Spectra for Complex Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2010.
- [26] O. M. Baksalary and G. Trenkler, "On K-potent matrices," *Electron. J. Linear Algebra*, vol. 26, pp. 446–470, 2013.
- [27] S. Boyd and L. Vandenberghe, *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares*. Cambridge, U.K.: Cambridge Univ. Press, 2018.
- [28] A. Zygmund, *Trigonometric Series, Vol. I and II*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 1968.

- [29] W. A. Sethares and T. W. Staley, "Periodicity transforms," *IEEE Trans. Signal Process.*, vol. 47, no. 11, pp. 2953–2964, Nov. 1999.
- [30] E. Andres, A. Barrat, and M. Karsai, "Detecting periodic time scales in temporal networks," 2023, *arXiv:2307.03840*.
- [31] J. Stehlé et al., "High-resolution measurements of face-to-face contact patterns in a primary school," *PLoS One*, vol. 6, no. 8, 2011, Art. no. e23176.
- [32] M. Génois and A. Barrat, "Can co-location be used as a proxy for face-to-face contacts?," *EPJ Data Sci.*, vol. 7, pp. 1–18, 2018.
- [33] P. Vanhems et al., "Correction: Estimating potential infection transmission routes in hospital wards using wearable proximity sensors," *PLoS One*, vol. 8, no. 9, 2013, Art. no. e73970.
- [34] M. Génois, C. Vestergaard, J. Fournet, A. Panisson, I. Bonmarin, and A. Barrat, "Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers," *Netw. Sci.*, vol. 3, no. 3, pp. 326–347, 2015.



Sergey Shvydun received the master's degree in business informatics (with distinction, 2014) and the Ph.D. degree in applied mathematics (*cum laude*, 2020) from the HSE University, Moscow, Russia. He has been a Postdoctoral Researcher with the Delft University of Technology, Delft, The Netherlands, since 2023. His research interests include network science, machine learning, operations research, and social choice theory. Before joining Delft, he was an Associate Professor with the HSE University and a Senior Research Fellow with the Institute of Control Sciences of the Russian Academy of Science, Moscow. He is the author of more than 25 papers in peer-reviewed journals and edited volumes and one book on centrality in networks.



Piet Van Mieghem (Fellow, IEEE) received the master's (*Magna cum Laude*) and Ph.D. degrees (*Summa cum Laude* with Congratulations) in electrical engineering from the K.U.Leuven, Leuven, Belgium, in 1987 and 1991, respectively. He is the Professor with the Delft University of Technology with a Chair in telecommunication networks and has been the Chairman of the section Network Architectures and Services (NAS) since 1998. His research interests include in the modelling and analysis of complex networks (such as infrastructural, biological, brain, social networks) and in new internet-like architectures and algorithms for future communications networks. The focus of the Chair is broadened from telecommunication networks to network science. He is the author of four books: Performance Analysis of Communications Networks and Systems, Data Communications Networking, and Graph Spectra for Complex Networks and Performance Analysis of Complex Networks and Systems. He is a board Member of the Netherlands Platform of Complex Systems, a steering committee Member of the Dutch Network Science Society, an external faculty Member with the Institute for Advanced Study (IAS), the University of Amsterdam. He was the Advanced ERC grant 2020 for ViSiON, Virus Spread in Networks. Before joining Delft, he was with the Interuniversity Micro Electronic Center (IMEC) from 1987 to 1991. During 1993 to 1998, he was a Member of the Alcatel Corporate Research Center, Antwerp, Belgium, where he was engaged in performance analysis of ATM systems and in network architectural concepts of both ATM networks (PNNI) and the internet. He was a visiting Scientist with Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA from 1992 to 1993 and a visiting Professor with Department of Electrical Engineering, University of California, Los Angeles, CA, USA, in 2005, Center of Applied Mathematics, Cornell University, Ithaca, NY, USA, in 2009, Department of Electrical Engineering, Stanford University, Stanford, CA, USA, in 2015, and Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ, USA, in 2022. He is on the editorial board of the *OUP Journal of Complex Networks*. He was the Member of the editorial board of *Computer Networks* (2005–2006), the *ACM/IEEE TRANSACTIONS ON NETWORKING* (2008–2012), the *Journal of Discrete Mathematics* (2012–2014), and *Computer Communications* (2012–2015).