Advances in Data Analysis and Classification
https://doi.org/10.1007/s11634-025-00634-9

REGULAR ARTICLE

Variational inference for estimating dynamic stochastic block models through an evolutionary algorithm

Luca Brusa[1] · **Fulvia Pennoni**[1]

Received: 18 December 2023 / Revised: 14 February 2025 / Accepted: 16 March 2025

Abstract
Dynamic temporal networks are important structures to capture node dependencies and their evolution over time. The dynamic stochastic block model, commonly used with longitudinal network data, is estimated maximizing the likelihood function through the variational expectation-maximization (VEM) algorithm. However, maximization is challenging due to the presence of multiple local maxima. In this paper, we first conduct a simulation study to assess the performance of six different parameter initialization strategies. Second, we introduce a novel specification of the VEM through a genetic algorithm, enabling a more comprehensive exploration of the parameter space. Results from both simulations and historical data on infectious disease transmission highlight the advantages of this approach in overcoming convergence to local maxima and improving node clustering in temporal network data.

Keywords Genetic algorithm · Local maxima · Longitudinal networks · Node classification · Variational expectation-maximization algorithm

Mathematics Subject Classification 62M05 · 62H30 · 65K10 · 91D30

1 Introduction

The stochastic block (SB) model (Snijders and Nowicki, 1997; Nowicki and Snijders, 2001) is tailored to analyze network data arising in various contexts, such as business, human resources, social media, and biological interactions. When dealing with cross-sectional network data, the model allows us to identify communities, represented as

✉ Luca Brusa
luca.brusa@unimib.it

Fulvia Pennoni
fulvia.pennoni@unimib.it

[1] Department of Statistics and Quantitative Methods, University of Milano-Bicocca, Via Bicocca degli Arcimboldi 8, 20126 Milan, Italy

Published online: 23 May 2025

groups of nodes that are strongly connected by edges, and clusters, which are groups of nodes exhibiting similar association patterns. The dynamic SB (DSB) model is proposed to analyze temporal network data, when multiple snapshots are observed at different time occasions; see Yang et al. (2011), Matias and Miele (2017), and Bartolucci and Pandolfi (2020) for additional details. In this context of network analysis, discrete latent variables are used for their flexibility in modeling complex dependence structures; see Bartolucci et al. (2022) for a comprehensive review of these models. Parameters of the DSB model are typically estimated using an approximated maximum likelihood approach with variational inference (Blei et al., 2017), implemented through a variational expectation-maximization (VEM) algorithm (Jordan et al., 1999; Daudin et al., 2008). The VEM maximizes a lower bound of the log-likelihood function, assuming that the latent variables are a posteriori independent either across times and subjects, as proposed in Yang et al. (2011), or across subjects only, as in Matias and Miele (2017). An exact formulation of the VEM algorithm for maximizing the target function is presented in Bartolucci and Pandolfi (2020), which avoid the use of approximate steps found in previous works, thereby improving clusters identifiability.

The multimodality of the log-likelihood function may induce the VEM algorithm to be trapped in one of the multiple local maxima. To address this issue, initialization procedures are commonly employed, though they are overlooked in the context of the DSB model. Evolutionary algorithms (EAs) constitute a family of computational techniques designed to tackle complex optimization problems; see, among others, Bäck (1996), Deb (2001), and Ashlock (2004). Through iterative cycles, EAs progressively refine candidate solutions by evaluating and selecting them, while applying controlled random variations to generate new potential solutions. EAs are widely used for unsupervised clustering, as discussed, among the others, by Hruschka et al. (2009), and for the estimation of Gaussian finite mixture models; see, for instance, Pernkopf and Bouchaffra (2005), Andrews and McNicholas (2013), and McNicholas et al. (2021).

Evolutionary versions of the EM and VEM algorithms (referred to as EEM and EVEM, respectively) are implemented in Brusa et al. (2024) to estimate other discrete latent variable (DLV) models, enhancing the exploration of the parameter space and facilitating convergence to the global maximum. Following this line of research, we propose a refined version of the EVEM algorithm for estimating the DSB model. We conduct an extensive Monte Carlo simulation study in which (*i*) we compare six different initialization strategies for the VEM algorithm, some of which have not previously been used for this model, and (*ii*) we evaluate the performance of the proposed EVEM algorithm to reach the global maximum and accurately recover the clustering structure. We also illustrate the feasibility of the proposed estimation method through an application using data on repeated face-to-face contacts between patients and healthcare workers in a hospital ward, aimed at investigating the transmission of an infectious disease (Vanhems et al., 2013). We also mention other approaches used to analyze the same data available in the literature. The code for our proposal is implemented in C++ and is available for use with R (R Core Team, 2025) at the following GitHub repository: https://github.com/LB1304/dynSBevo. A parallelized version of the algorithm is also provided to enhance computational efficiency and scalability.

The remainder of this paper is organized as follows. Section 2 introduces the DSB model and reviews the estimation procedure through the VEM algorithm. Section 3 describes the proposed EVEM algorithm. Section 4 presents simulation results that assess the performance of the VEM algorithm under various initializations and compare the VEM and EVEM algorithms. Section 5 demonstrates the EVEM algorithm performance in estimating the DSB model with data on face-to-face contacts in a hospital ward. Finally, Sect. 6 provides concluding remarks. Additional details regarding the simulation scenarios, the parallelization of the code, the related computational time, and the simulation results are provided in the Appendices A, B, C, and D, respectively.

2 Binary dynamic stochastic block model

In the following, we introduce the notation of the binary DSB model and present the approximate maximum likelihood estimation procedure implemented through the VEM algorithm.

2.1 Notation

Let $Y_{ij}^{(t)}$ be a binary variable indicating the presence of an edge between nodes i and j, where $i, j = 1, \ldots, n$ and $i \neq j$, at time t, with $t = 1, \ldots, T$. Let $\boldsymbol{Y}$ denote the adjacency array of dimensions $n \times n \times T$, where $\boldsymbol{Y}^{(t)}$ is the binary adjacency matrix at time t. We consider an undirected network without self-loops, so that $Y_{ij}^{(t)} = Y_{ji}^{(t)}$ and $Y_{ii}^{(t)} = 0$. Nodes are distributed in a finite set of k blocks, defined by a latent process $\boldsymbol{Z}_i = (Z_i^{(1)}, \ldots, Z_i^{(T)})'$, which is specific to each node i. The collection of these processes is represented as $\boldsymbol{Z} = (\boldsymbol{Z}_i)_{i=1,\ldots,n}$, and we assume that $\boldsymbol{Z}$ follows a Markov chain of first-order. The initial and transition probabilities define the block proportions at time $t = 1$ and the probabilities of switching between blocks over time for $t = 2, \ldots, T$, respectively. These probabilities are denoted as $\alpha_u = p(Z_i^{(1)} = u)$ for $u = 1, \ldots, k$, and $\pi_{uv} = p(Z_i^{(t)} = v | Z_i^{(t-1)} = u)$ for $u, v = 1, \ldots, k$.

Variables $Y_{ij}^{(t)}$ are assumed to be locally independent and identically distributed as Bernoulli random variables, conditional on the latent blocks to which nodes i and j belong at time t:

$$Y_{ij}^{(t)} | \{Z_i^{(t)} = u, Z_j^{(t)} = v\} \sim \mathcal{B}(\beta_{uv}),$$

where $\beta_{uv} = p(Y_{ij}^{(t)} = 1 | Z_i^{(t)} = u, Z_j^{(t)} = v)$ for $u, v = 1, \ldots, k$, are referred to as connection probabilities.

Let $y_{ij}^{(t)}$ denote a realization of $Y_{ij}^{(t)}$ and $z_i^{(t)}$ a realization of $Z_i^{(t)}$. The conditional distribution of the response variables given the latent variables and the probability

mass function of the latent variables are defined as follows

$$p(\boldsymbol{Y}|\boldsymbol{Z}) = \prod_{t=1}^{T}\prod_{i=1}^{n}\prod_{j=i+1}^{n} p(y_{ij}^{(t)}|z_i^{(t)}, z_j^{(t)}),$$

$$p(\boldsymbol{Z}) = \prod_{i=1}^{n} \alpha_{z_i^{(1)}} \prod_{t=2}^{T} \pi_{z_i^{(t-1)} z_i^{(t)}}.$$

The observed network mass function is obtained through suitable recursions based on the previous assumptions, and is given by

$$p(\boldsymbol{Y}) = \sum\nolimits_{\mathcal{Z}} p(\boldsymbol{Y}|\boldsymbol{Z}) p(\boldsymbol{Z}),$$

where $\sum_{\mathcal{Z}}$ denotes the sum over the support of $\boldsymbol{Z}$. For more details on this model formulation, see Bartolucci and Pandolfi (2020).

2.2 Variational EM algorithm

Maximum likelihood inference within DLV models is typically carried out using the EM algorithm (Dempster et al., 1977). This algorithm maximizes the expectation of the complete-data log-likelihood function of the model, relying on the conditional distribution of the latent variables given the data, denoted as $p(\boldsymbol{Z}|\boldsymbol{Y})$. When dealing with the SB and DSB models, however, this distribution is intractable. Indeed, as shown in Matias and Robin (2014) with reference to a static network, the latent variables Z_i, Z_j are not independent given Y_{ij}. Consequently, the conditional distribution cannot be factorized, making the computation prohibitive even for the simplest graphs. Daudin et al. (2008) propose to overcome this problem by maximizing an approximation of the log-likelihood function defined through a variational distribution using the VEM algorithm. Their proposal was extended in Yang et al. (2011) to the context of DSB models. More specifically, this approach relies on a suitable approximation of the conditional distribution $p(\boldsymbol{Z}|\boldsymbol{Y})$, denoted as $q(\boldsymbol{Z})$, and defines the following function:

$$\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau}) = \mathbb{E}_q[\log p(\boldsymbol{Y}, \boldsymbol{Z})] - \mathbb{E}_q[\log q(\boldsymbol{Z})],$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\tau}$ collect the model and the variational parameters, respectively. Denoting by KL$[\cdot||\cdot]$ the Kullback–Leibler divergence (Kullback and Leibler, 1951), it is straightforward to prove that

$$\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau}) = \log p(\boldsymbol{Y}) - \mathrm{KL}[q(\boldsymbol{Z})||p(\boldsymbol{Z}|\boldsymbol{Y})], \tag{1}$$

thus showing that $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$ constitutes a lower bound of the observed-data log-likelihood function.

As highlighted in Sect. 1, different variational approximations have been proposed. Considering the one introduced in Matias and Miele (2017), it results as:

$$q(\boldsymbol{Z}) = \prod_{i=1}^{n}\left[q(Z_i^{(1)}) \prod_{t=2}^{T} q(Z_i^{(t)}|Z_i^{(t-1)}) \right]$$
$$= \prod_{i=1}^{n}\left[\prod_{u=1}^{k} \tau(i,u)^{I(Z_i^{(1)}=u)} \prod_{t=2}^{T}\prod_{u=1}^{k}\prod_{v=1}^{k} \tau(t,i,u,v)^{I(Z_i^{(t-1)}=u)I(Z_i^{(t)}=v)} \right],$$

with $I(\cdot)$ denoting the indicator function. In the above expression, the variational parameters $\tau(i,u)$ and $\tau(t,i,u,v)$ are interpreted as approximations of $p(Z_i^{(1)} = u|\boldsymbol{Y})$ and $p(Z_i^{(t)} = v|Z_i^{(t-1)} = u, \boldsymbol{Y})$, respectively. In this case, Equation (1) may be written as

$$\begin{aligned}\mathcal{J}(\boldsymbol{\theta},\boldsymbol{\tau}) = &\sum_{i=1}^{n}\sum_{u=1}^{k} \tau(i,u)\big[\log\alpha_u - \log\tau(i,u)\big] \\ &+ \sum_{t=2}^{T}\sum_{i=1}^{n}\sum_{u=1}^{k}\sum_{v=1}^{k} \tau_m(t-1,i,u)\tau(t,i,u,v)\big[\log\pi_{uv} - \log\tau(t,i,u,v)\big] \\ &+ \sum_{t=1}^{T}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{u=1}^{k}\sum_{v=1}^{k} \tau_m(t,i,u)\tau_m(t,j,v)\log\phi(y_{ij}^{(t)},\beta_{uv}),\end{aligned}$$

where $\phi(y_{ij}^{(t)},\beta_{uv}) = \beta_{uv}^{y_{ij}^{(t)}}(1-\beta_{uv})^{1-y_{ij}^{(t)}}$ and

$$\tau_m(t,i,u) = \begin{cases} \tau(i,u) & t = 1, \\ \sum_{v=1}^{k} \tau_m(t-1,i,v)\tau(t,i,v,u) & t = 2,\ldots,T. \end{cases}$$

The VEM algorithm consists in alternating updates of the variational ($\boldsymbol{\tau}$) and the model ($\boldsymbol{\theta}$) parameters until convergence. In more details, the two steps are:

(i) A variational expectation step, in which $\mathcal{J}(\boldsymbol{\theta},\boldsymbol{\tau})$ is maximized with respect to $\boldsymbol{\tau}$ in order to refine the approximation of the log-likelihood;
(ii) A maximization step, in which $\mathcal{J}(\boldsymbol{\theta},\boldsymbol{\tau})$ is maximized with respect to $\boldsymbol{\theta}$, thus updating the model parameters.

Convergence of the VEM algorithm is checked applying two common criteria, which require that both the relative difference of $\mathcal{J}(\boldsymbol{\theta},\boldsymbol{\tau})$ at two consecutive steps and the distance between the corresponding parameters are smaller than suitable thresholds.

3 Evolutionary VEM algorithm

In the following, we illustrate our proposal and highlight some initialization strategies for the VEM algorithm.

3.1 Steps of the EVEM algorithm

The EVEM algorithm proposed in Brusa et al. (2024) for the estimation of the static SB model considers an initial population P_0 of candidate solutions, generated through a random process. It incorporates two distinct sources of variation: crossover, to generate new potential solutions from the existing ones, and mutation, to introduce modifications to single candidates. Best solutions are selected for the successive step of the iterative procedures, while the others are discarded. The EVEM algorithm proposed here for estimating the DSB model is implemented according to the following specific features: (*i*) an initial population P_0 of candidate solutions for the estimation problem at hand, here specified as possible arrays for the variational parameters $\tau(i, t)$ and $\tau(t, i, u, v)$ defined in Sect. 2.2. Since random initialization is often inappropriate for DSB models, the current proposal compares several other initialization strategies for the starting population P_0, as discussed in Sect. 4.1; (*ii*) a mutation operator that introduces variations to the existing candidates and generates new solutions by randomly selecting an observation and changing its cluster membership. Generalizing the proposal of Brusa et al. (2024), each candidate solution is subjected to the mutation operator with its own probability p_m, which may differ from those assigned to other candidates. Specifically, p_m is randomly drawn from a uniform distribution $U(a, b)$, with $0 < a \leq b < 1$. Crossover is not considered at all; (*iii*) selection of the best solutions based on a quality measure that favors candidates with higher values of $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$.

The initial population, is defined as $P_0 = P_0^R \cup P_0^{KM} \cup P_0^{SBM}$, where P_0^R, P_0^{KM}, and P_0^{SBM} are sets of $N/3$ candidate solutions each. In particular, elements in population P_0^R are obtained through a random initialization. For populations P_0^{KM} and P_0^{SBM}, the first candidate is selected according to the k-means and WgtSBM initializations, respectively, as defined in the following Sect. 3.2. The remaining $N/3 - 1$ candidates are obtained through mutation. Note that, although we consider $|P_0^R| = |P_0^{KM}| = |P_0^{SBM}| = N/3$, this approach is highly flexible and can be easily generalized by considering different cardinalities or new initialization strategies.

In more details, the pseudo-code of the algorithm is summarized in Algorithm 1. The procedure alternates the following steps until convergence:

1. *Update:* Population P_0 is updated through R iterations of the VEM algorithm applied to each individual, resulting in an updated population P_1. To ensure computational efficiency, the value of R is kept sufficiently small. Convergence is determined by the relative change of the log-likelihood between successive steps. If this condition is met, the VEM algorithm is stopped to avoid unnecessary additional cycles.
2. *Mutate:* To encourage a broader exploration of the parameter space, variation is introduced to each individual of population P_1 with a certain probability $p_m \sim U(a, b)$, where $0 < a \leq b < 1$. The new generation of mutated individuals is denoted as P_2.

3. *Update:* Population P_2 is updated through R iterations of the VEM algorithm applied to each individual, resulting in the generation of population P_3. This procedure replicates the previous update executed in Step 1.
4. *Select:* Individuals of both populations P_1 and P_3 are considered jointly, and the set of N individuals showing the highest value of $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$ are retained for the next generation, denoted as P_4.

Convergence is evaluated by examining the best solution from population P_4 at each step and analyzing the relative differences of $\mathcal{J}(\boldsymbol{\theta}, \boldsymbol{\tau})$ between two consecutive iterations, as well as differences between corresponding parameter vectors. In summary, the stopping criterion for the algorithm is met when

$$\frac{|\mathcal{J}(\boldsymbol{\theta}^{(h)}, \boldsymbol{\tau}^{(h)}) - \mathcal{J}(\boldsymbol{\theta}^{(h-1)}, \boldsymbol{\tau}^{(h-1)})|}{|\mathcal{J}(\boldsymbol{\theta}^{(h)}, \boldsymbol{\tau}^{(h)})|} < \varepsilon_1 \quad \text{and} \quad \max_s |\theta_s^{(h)} - \theta_s^{(h-1)}| < \varepsilon_2,$$

where $\boldsymbol{\theta}^{(h)}$ denotes the estimated parameter vector corresponding to the optimal candidate solution during the h-th iteration of the EVEM algorithm. In this study, the tolerance levels ε_1 and ε_2 are set to 10^{-8}.

Algorithm 1 General scheme of the EVEM algorithm

Algorithm 1 General scheme of the EVEM algorithm

Initialize: N, R, p_m, and $P_0 = P_0^R \cup P_0^{KM} \cup P_0^{SBM}$
while (Convergence condition = FALSE) **do**
    $P_1 \leftarrow$ Update(P_0): run R steps of the VEM algorithm
    $P_2 \leftarrow$ Mutate(P_1)
    $P_3 \leftarrow$ Update(P_2): run R steps of the VEM algorithm
    $P_4 \leftarrow$ Select($P_1 \cup P_3$)
    $P_0 \leftarrow P_4$
end while
Select the best result from population P_4

3.2 Initialization strategies

In the following, we present several proposals to initialize the variational parameters $\tau(i, u)$ and $\tau(t, i, u, v)$ for $t = 1, \ldots, T$, $i = 1, \ldots, n$, and $u, v = 1, \ldots, k$ described in Sect. 2.2. In particular, the following six strategies are employed to obtain an initial partition of nodes into k latent blocks; the starting values are then computed on the basis of this partition.

A. *Random sampling:* A simple initialization approach in which each node is randomly assigned to one of the k latent blocks at each time point $t = 1, \ldots, T$.
B. *Clustering algorithms:* A widely adopted approach, common to other DLV models, for generating an initial clustering. Nodes are grouped by applying a clustering algorithm to the rows of an extended matrix created by concatenating the adjacency matrices $\boldsymbol{Y}^{(t)}$, $t = 1, \ldots, T$. Here we explore:

(B_1) k-means algorithm (MacQueen, 1967), widely employed to initialize the EM and VEM algorithms for many DLV models;
(B_2) k-medoids clustering (Kaufman and Rousseeuw, 1990), a more robust alternative to k-means;
(B_3) Spectral clustering (Von Luxemburg, 2007), frequently and effectively employed to initialize the VEM algorithm in the estimation of the static SB model. Here, we consider global spectral clustering (dynSC) proposed by Liu et al. (2018).

C. *Network models:* This strategy involves initializing the DSB model through the estimated parameters of a static SB model fit on the data aggregated across all time occasions. We consider the following two methods:

(C_1) Binarized network (BinSBM): the static SB model is applied to the fully binarized aggregated network $\boldsymbol{Y}_{\text{Agg}}$, where edges are considered as either present or absent, without accounting for temporal differences. Specifically, $\boldsymbol{Y}_{\text{Agg}} = 1$ if there exists at least one t (for $t = 1, \ldots, T$) such that $\boldsymbol{Y}^{(t)} = 1$; otherwise, $\boldsymbol{Y}_{\text{Agg}} = 0$.
(C_2) Weighted network (WgtSBM): the aggregated network retains a weighted representation to account for strength of interactions. Specifically, $\boldsymbol{Y}_{\text{Agg}} = \sum_{t=1}^{T} \boldsymbol{Y}^{(t)}$.

4 Simulation study

We present the results of an extensive Monte Carlo simulation study designed to (*i*) compare the performance of the six initialization strategies illustrated in Sect. 3.2, and (*ii*) evaluate the efficacy of the proposed EVEM algorithm for estimating the DSB model. We consider sixty-four different scenarios defined by varying the number of nodes ($n = 20, 50, 100$), latent blocks ($k = 2, 3, 5$), and time occasions ($T = 6, 10$), with different block persistence (high or low), and connectivity parameters (intra-group greater or smaller than inter-group). For more details on the simulated settings for each scenario, please refer to Appendix A.

4.1 Evaluation of the initialization strategies

We assess the performance of the six aforementioned initializations, which are implemented in R using the following functions: (B_1) `kmeans` from the `stats` package for the k-means algorithm; (B_2) `pam` from the `cluster` package (Maechler et al., 2023) for the k-medoids algorithm; (C_1 and C_2) `estimateSimpleSBM` from the `sbm` package (Chiquet et al., 2024) for the BinSBM and WgtSBM approaches. For each initialization strategy, we evaluate the Adjusted Rand Index (ARI, Hubert and Arabie, 1985) and the frequency of convergence to the global maximum obtained under the 64 simulated scenarios illustrated in Appendix A. Tables 1 and 2 provide a detailed

Table 1 Average ARI values between simulated and estimated clusters over 100 samples for each initialization strategy of the VEM algorithm, across a selection of scenarios outlined in Appendix A, with $n = 50$ and $T = 6$. The columns labeled "High" and "Low" indicate cases with high and low probabilities of block persistence (π_{uu}), respectively. Values in bold highlight the best results

		$\beta_{uu} > \beta_{uv}$		$\beta_{uu} < \beta_{uv}$	
		π_{uu}			
	Initialization	High	Low	High	Low
$k = 2$	Random	0.750	0.518	0.724	0.376
	k-means	**1.000**	0.966	**0.993**	0.912
	k-medoids	0.989	0.607	0.830	0.464
	DynSC	0.248	0.210	0.206	0.150
	BinSBM	0.989	0.957	0.990	0.911
	WgtSBM	0.990	**0.993**	0.990	**0.929**
$k = 3$	Random	0.622	0.499	0.546	0.340
	k-means	0.888	0.702	0.897	0.639
	k-medoids	0.731	0.410	0.646	0.371
	DynSC	0.172	0.173	0.198	0.169
	BinSBM	**0.969**	0.745	0.950	0.616
	WgtSBM	0.963	**0.784**	**0.956**	**0.652**
$k = 5$	Random	0.506	0.357	0.320	0.055
	k-means	0.689	0.493	0.535	**0.155**
	k-medoids	0.594	0.374	0.418	0.079
	DynSC	0.157	0.150	0.237	0.064
	BinSBM	0.781	0.547	0.651	0.118
	WgtSBM	**0.830**	**0.584**	**0.679**	0.112

comparison for scenarios with $n = 50$ and $T = 6$. Additionally, Table 5 in Appendix D shows further results for all 64 scenarios.

The following findings emerge from the simulation results:

- The k-means, WgtSBM, and BinSBM methods provide the best results across all simulated scenarios. The two approaches based on the static SB model, in particular, exhibit similar behavior. To our knowledge, we are the first to use this initialization approach, which appears most suitable, particularly for DSB models with many latent blocks;
- Random initialization exhibits poor performance in the context of DSB models, as also evidenced in Matias and Miele (2017). This finding has been important for the implementation of the EVEM algorithm, as discussed in Sect. 3.1. Similarly, the k-medoids method yields suboptimal solutions compared to the k-means approach;

Table 2 Sample proportion converging to the global maximum over 100 samples for each initialization strategy of the VEM algorithm, across a selection of scenarios outlined in Appendix A, with $n = 50$ and $T = 6$. The columns labeled "High" and "Low" indicate cases with high and low probabilities of block persistence (π_{uu}), respectively. Values in bold highlight the best results

		$\beta_{uu} > \beta_{uv}$		$\beta_{uu} < \beta_{uv}$	
		π_{uu}			
	Initialization	High	Low	High	Low
$k = 2$	Random	0.750	0.410	0.730	0.350
	k-means	**1.000**	**0.940**	**1.000**	0.910
	k-medoids	0.990	0.490	0.800	0.380
	DynSC	0.100	0.040	0.080	0.050
	BinSBM	0.520	0.860	0.880	0.900
	WgtSBM	0.530	0.910	0.880	**0.930**
$k = 3$	Random	0.300	0.180	0.270	0.140
	k-means	0.730	0.440	0.840	**0.540**
	k-medoids	0.380	0.080	0.360	0.110
	DynSC	0.000	0.050	0.020	0.090
	BinSBM	**0.750**	0.520	**0.860**	0.370
	WgtSBM	0.690	**0.540**	0.820	0.410
$k = 5$	Random	0.040	0.080	0.060	0.100
	k-means	0.200	**0.360**	0.280	**0.280**
	k-medoids	0.080	0.080	0.140	0.220
	DynSC	0.000	0.000	0.100	0.200
	BinSBM	0.440	0.300	0.400	0.180
	WgtSBM	**0.640**	0.280	**0.600**	0.200

- DynSC initialization performs poorly in this context, even though it demonstrates optimal performance for the static SB model, as illustrated in Gao et al. (2017).

4.2 Evaluation of VEM and EVEM algorithms

In the following, we compare the performance of the proposed EVEM algorithm with the standard VEM algorithm. For each simulated scenario summarized in Appendix A, we randomly draw 100 samples of size n, estimating the DSB model using both approaches. In particular, for the VEM algorithm, we employ different initialization strategies: k-means, BinSBM, and WgtSBM, as suggested by the results in Sect. 4.1, along with 20 runs of random initialization. The EVEM algorithm is initialized according to the approach described in Sect. 3.1.

Table 3 Average ARI values between simulated and estimated clusters over 100 samples using the EVEM algorithm across 64 simulated scenarios. The columns labeled "High" and "Low" indicate cases with high and low probabilities of block persistence (π_{uu}), respectively. Values in parentheses indicate the increase or decrease obtained with the EVEM compared to the VEM algorithm

			$\beta_{uu} > \beta_{uv}$		$\beta_{uu} < \beta_{uv}$	
			π_{uu}			
			High	Low	High	Low
$k = 2$	$n = 20$	$T = 6$	0.978 (+0.009)	0.820 (+0.040)	0.906 (+0.002)	0.635 (+0.142)
		$T = 10$	0.966 (−0.010)	0.721 (+0.153)	0.904 (+0.002)	0.544 (+0.159)
	$n = 50$	$T = 6$	1.000 (0.000)	0.999 (0.000)	0.993 (0.000)	0.957 (+0.016)
		$T = 10$	1.000 (0.000)	0.959 (+0.052)	0.995 (0.000)	0.850 (+0.033)
	$n = 100$	$T = 6$	1.000 (0.000)	1.000 (0.000)	1.000 (0.000)	1.000 (0.000)
		$T = 10$	1.000 (0.000)	0.945 (+0.013)	1.000 (0.000)	0.945 (+0.035)
$k = 3$	$n = 20$	$T = 6$	0.739 (+0.040)	0.371 (+0.099)	0.650 (+0.087)	0.201 (+0.052)
		$T = 10$	0.853 (+0.076)	0.371 (+0.109)	0.706 (+0.109)	0.149 (+0.045)
	$n = 50$	$T = 6$	0.989 (+0.006)	0.953 (+0.076)	0.972 (+0.011)	0.883 (+0.116)
		$T = 10$	0.993 (+0.019)	0.891 (+0.133)	0.974 (+0.006)	0.776 (+0.171)
	$n = 100$	$T = 6$	0.995 (−0.002)	0.992 (+0.045)	0.996 (+0.004)	0.970 (+0.043)
		$T = 10$	1.000 (+0.024)	0.972 (+0.106)	0.980 (+0.002)	0.955 (+0.114)
$k = 5$	$n = 50$	$T = 6$	0.886 (+0.077)	0.710 (+0.136)	0.743 (+0.062)	0.224 (+0.071)
		$T = 10$	0.872 (+0.081)	0.753 (+0.240)	0.796 (+0.091)	0.153 (+0.097)
	$n = 100$	$T = 6$	0.950 (+0.041)	0.918 (+0.106)	0.975 (+0.004)	0.928 (+0.103)
		$T = 10$	0.967 (+0.082)	0.891 (+0.180)	0.979 (+0.007)	0.888 (+0.083)

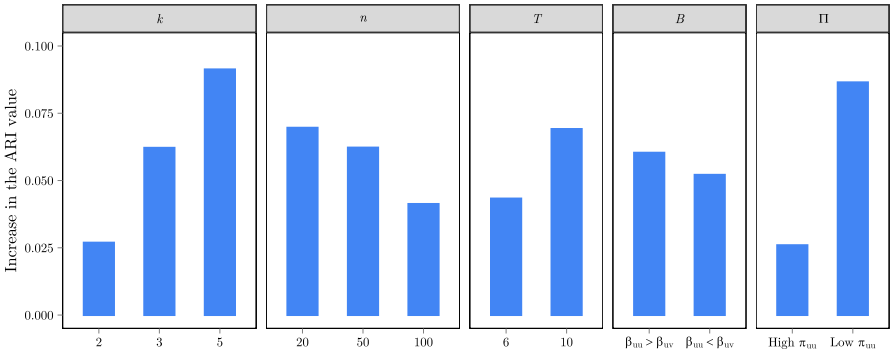


Fig. 1 Average ARI gain achieved by the EVEM algorithm compared to the VEM algorithm for each model specification in the simulation scenarios, varying the number of latent blocks (k), of nodes (n), and of time occasions (T), the connectivity parameters in matrix $\boldsymbol{B}$, and the transition probabilities in matrix $\boldsymbol{\Pi}$

Table 3 shows the comparison of ARI values between simulated and estimated clustering. Each value represents the average across simulated samples within each scenario. The EVEM algorithm consistently demonstrates superior classification performance across nearly all scenarios, achieving the highest ARI in the majority of cases. Only in two scenarios does the standard VEM algorithm slightly outperform EVEM; however, in these cases, the ARI is close to 1 for both algorithms. Both the VEM and EVEM algorithms show notably low ARI values in cases with few nodes and many latent blocks, such as scenarios with $n = 20$ and $k = 3$ or $n = 50$ and $k = 5$. We further explore the conditions under which the EVEM algorithm provides the most significant improvements. Figure 1 illustrates the average ARI gain achieved by the EVEM algorithm compared to the VEM algorithm for each model specification in the simulated scenarios.

The results indicate that the benefits of using the EVEM algorithm increase substantially with the number of latent blocks; for $k = 2$, the average ARI improvement is approximately 0.027, rising to nearly 0.100 for $k = 5$. A similar trend is observed with varying levels of transition matrix persistence: under high persistence, the ARI gain is limited to around 0.026, while with low persistence, it reaches an average improvement of 0.087. In contrast, there is no significant difference in performance based on connectivity behavior, with the ARI gain remaining around 0.060 in both high intra-group and high inter-group connectivity cases. Lastly, the superior classification performance of the EVEM algorithm becomes most evident as the number of time points increases and the number of nodes decreases. Table 6 in Appendix D also shows the sample proportion in which the log-likelihood value at convergence, achieved through the EVEM algorithm, is higher than the corresponding value obtained using the VEM algorithm. This proportion is large across all scenarios, particularly under the most complex DSB models.

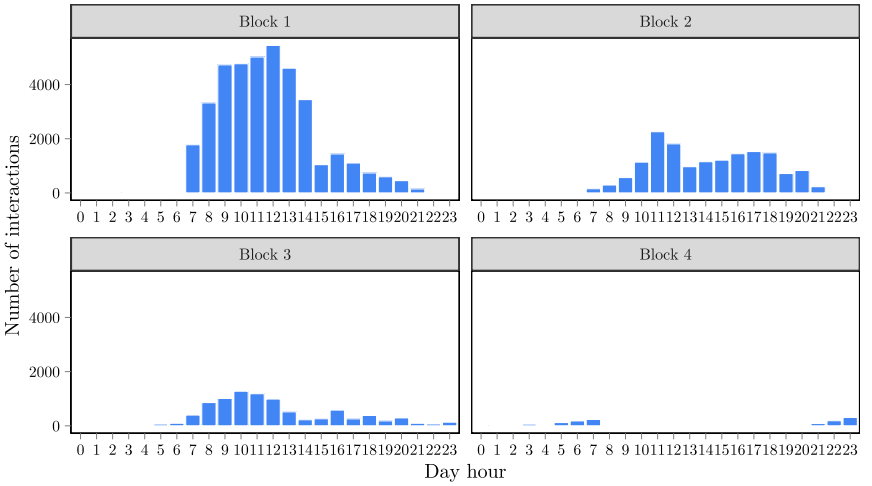
5 Application

Data refer to face-to-face contacts detected by wearable sensors among $n = 75$ patients and healthcare workers in a hospital ward in Lyon, France (Vanhems et al., 2013). Data were collected over five working days ($T = 5$), from December 6 to December 10, 2010,[1] to analyze the spread of an infectious disease within the ward. The sample consist of patients, medical doctors, nurses, and administrative staff, consisting of 29, 11, 27, and 8, respectively. The binary T-dimensional adjacency array $\boldsymbol{Y}$ is constructed by setting each element $Y_{ij}^{(t)}$ equal to 1 if at least one face-to-face contact is recorded between individuals i and j at time t.

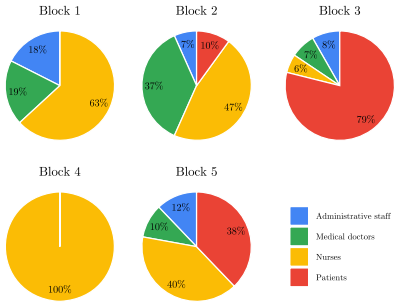
A DSB model with 5 latent blocks is estimated using both the VEM and the EVEM algorithms. In particular, as discussed in Sect. 4, we apply different initialization strategies for the VEM algorithm (namely, k-means, BinSBM, and WgtSBM), along with 20 runs of random initialization. The EVEM algorithm is initialized as illustrated in Sect. 3.1. The superior performance of the evolutionary algorithm demonstrated in the simulation study is also confirmed in this example. In fact, the value of $\mathcal{J}(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\tau}})$ at convergence increases from $-2{,}915$ to $-2{,}902$ passing from the VEM to the EVEM algorithm. Interestingly, for the VEM algorithm, the best results are obtained with random initialization, while the $k-$means, BinSBM, and WgtSBM approaches perform quite poorly on these data. This highlights the utility of relaying on the EVEM algorithm with sample data.

The estimated parameters of the DSB model suggest interpreting the latent blocks primarily based on the contacts that occurred among patients and healthcare workers. Table 4 summarizes the results, showing the average number of connections among the allocated nodes for each time occasion $t = 1, \ldots, 5$ and each latent block $u = 1, \ldots, 5$. Furthermore, Fig. 2 (panels a and b) reports the number of interactions during the hours of the day for each latent state and the proportion of administrative staff, medical doctors, nurses, and patients in each latent block, respectively. The 1st latent block is characterized by the highest average number of contacts. It is predominantly composed of nurses (around 63%), with the remaining individuals being medical and administrative personnel. This block likely represents core staff engaged in intensive interactions, primarily during daytime shifts. The internal connection probability is very high (0.95), and there are also strong interactions with the 2nd (0.72) and 3rd (0.43) blocks. The 2nd latent block consists mainly of nurses and doctors, who together make up approximately 83% of the group. This block exhibits fewer interactions compared to the first, particularly in terms of internal connections, with a probability of 0.38. However, similarly to the 1st block, interactions occur mainly during daytime hours. The 3rd latent block is primarily composed of patients (around 79%), with internal connections being almost nonexistent. The few interactions that do occur happen mainly in the morning and are mostly with the 1st block. The 4th latent state consists solely of nurses working night-shifts. Their interactions are primarily internal, with some connections to patients in the 3rd block. The 5th block includes individuals from all typologies, with a majority of nurses and patients. It is characterized by almost no

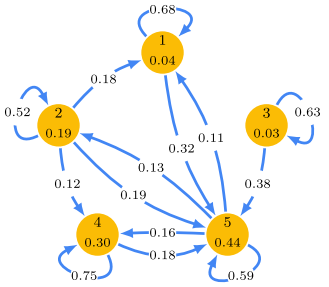
[1] Data are available at the website: http://www.sociopatterns.org/datasets/hospital-ward-dynamic-contact-network/ (Accessed in October 2024).



(a) Number of interactions during daytime hours for each latent block: an interaction is counted for block u if at least one node involved in the interaction belongs to block u. The 5th latent block is not reported, as the number of interactions is negligible



(b) Proportion of administrative staff, medical doctors, nurses, and patients in each latent block



(c) Averaged transition probabilities ($\hat{\pi}_{u|\bar{u}}$, along the edges) and initial probabilities ($\hat{\alpha}_u$, within the blocks)

Fig. 2 Summary of the estimated results for the hospital ward contact data under the DSB model with $k = 5$

interactions, likely reflecting patients in isolation or medical and administrative staff who were not working on the observed day. The estimated averaged transition matrix, depicted in Fig. 2 (panel c), illustrates evolving interaction patterns among patients and healthcare workers. The 1st, 3rd and 4th blocks show high persistence over time, with moderate transitions towards the 5th block.

Table 4 Average number of incident edges for the nodes in each latent block at each time occasion under the estimated DSB with $k = 5$ for the hospital ward contact data

u	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
1	21.000	31.430	30.500	29.150	26.77
2	11.200	20.000	19.170	18.750	15.86
3	11.500	10.000	13.000	15.500	11.00
4	4.680	11.400	10.520	9.410	6.86
5	0.030	0.000	0.000	0.080	0.03

5.1 Comparison with other approaches

In the following, we briefly reference and discuss some other relevant studies that have utilized the same data. Addressing a different research question, Wijayanto and Murata (2019) focus on the challenge of controlling epidemic spread in dynamic networks and developing effective protection strategies. They propose a multiple-turn approach that dynamically allocates protection based on the evolving network structure, using reinforcement learning and graph embedding techniques to identify and safeguard critical nodes over time. Results from these study demonstrate that this approach effectively mitigates epidemic spread: by removing incident edges from 15% of nodes, it preserves around 70% of the network from infection, assuming two popular epidemic models, susceptible-infectious-susceptible (SIS) and susceptible-infectious-recovered (SIR).

The work of Süveges and Olhede (2023) focuses on edge clustering, modeling temporal dependencies in dynamic networks by applying a generalized linear model to capture time-varying interactions and correlations between edges. The application of the model to hospital interaction data identifies six distinct edge clusters, based on contact probabilities, the time of day they occur, and the roles of the involved individuals. The two clusters with the highest contact probabilities are predominantly linked to interactions among doctors and nurses, with distinct daily patterns. Nurse-patient interactions are more varied, with certain clusters showing peaks in the morning and afternoon, reflecting the hospital shift structure. Patients exhibit much lower contact probabilities, particularly in interactions with other patients. Although a direct comparison with the latent blocks obtained with the approach proposed in this article is not straightforward, these results align with those obtained with our method, particularly in confirming the strong connections among doctors and nurses and the significantly lower contact probabilities for patients.

In the analysis performed using the model proposed by Olivella et al. (2022), clustering is applied to the nodes instead of the edges, resulting in the identifications of three clusters. The first cluster, characterized by the highest contact probabilities, is primarily composed of administrative staff, medical staff, and nurses. The second cluster exhibits moderate contact probabilities, while the third cluster, which is predominantly made up of patients, shows the lowest contact probabilities. However, unlike our model, this approach does not facilitate the differentiation of night-shift

nurses or the separation of patients with moderate interactions (the 3rd latent block in our results) from those with minimal interactions (the 5th latent block in our results).

6 Concluding remarks

In the field of dynamic stochastic block (DSB) model, aimed at describing interactions within a target population using temporal network data, we address the critical issue of convergence to local maxima when approximate maximum likelihood is employed through the variational expectation-maximization (VEM) algorithm. We propose six different initialization approaches and compare their effectiveness through a simulation study conducted under various designs, focusing on achieving a global maximum and recovering the simulated clustering structure. Following the proposal in Brusa et al. 2024, we introduce an evolutionary variational expectation-maximization (EVEM) algorithm. This approach utilizes an iterative procedure that refines solutions by evaluating and selecting candidates, while introducing random variations to generate novel potential solutions. To assess the performance of the proposed EVEM algorithm, we conduct an extensive Monte Carlo simulation study. Additionally, we analyze observed face-to-face contacts in a hospital ward, investigating interactions among groups of individuals over a five-day period to explore the dynamics of infectious disease spread within the ward. Results indicate that three initialization methods, one based on the k-means approach and the other two on the static (binary or weighted) stochastic block model, are the most effective. Furthermore, the EVEM algorithm consistently outperforms the standard VEM in nearly all cases.

It is important to note that several extensions could be explored along this research line. While this study focuses on undirected dynamic networks without self-loops, the proposed approach can be generalized to other data structures. Additionally, the proposed EVEM algorithm may be readily extended to estimate the DSB model for weighted dynamic networks.

Another relevant research direction involves the selection of the number of latent blocks, an aspect often overlooked in the context of DSB models. To address this, a potential refinement would be to update the evolutionary selection step by employing an information criterion as the fitness function, instead of the log-likelihood function. Finally, given the increasing prevalence of large-scale temporal networks, it is important to address the inherent scalability limitations of the stochastic block model; these limitations exist independently of the estimation algorithm used. As a first step toward addressing these challenges, we have implemented a parallelized version of the EVEM algorithm, which significantly enhances computational efficiency and supports larger-scale applications. See Appendix C for additional details. Future research will focus on developing even more efficient approaches to further optimize performance and scalability for complex networks.

Appendix

Appendix A illustrates the design adopted for the simulation study presented in Sect. 4, Appendix B offers a comparison of the computational time required to estimate the dynamic stochastic block (DSB) model using the VEM algorithm across different initialization strategies. Appendix C discusses the computational time of the proposed EVEM algorithm and introduces the parallelized implementation. Appendix D presents additional simulation results.

A Description of the simulation design

The design of the simulation study is built on samples of sizes $n = 20, 50, 100$ with $k = 2, 3, 5$ latent blocks, and $T = 6, 10$ time occasions. For the model parameters, the initial probability vector $\boldsymbol{\alpha}$ is randomly selected from a uniform distribution ($\alpha_u \sim U(0, 1)$) and appropriately normalized. Additionally, transition probability and connectivity matrices $\boldsymbol{\Pi}$ and $\mathbf{B}$ are fixed according to the design proposed by Bartolucci and Pandolfi 2020, featuring high and low values as follows:

- High persistent blocks (High π_{uu}):

$$\boldsymbol{\Pi} = \begin{bmatrix} 0.910 & 0.090 \\ 0.090 & 0.910 \end{bmatrix}, \quad \boldsymbol{\Pi} = \begin{bmatrix} 0.900 & 0.090 & 0.010 \\ 0.080 & 0.840 & 0.080 \\ 0.010 & 0.090 & 0.900 \end{bmatrix},$$

$$\boldsymbol{\Pi} = \begin{bmatrix} 0.880 & 0.080 & 0.040 & 0.000 & 0.000 \\ 0.060 & 0.840 & 0.060 & 0.040 & 0.000 \\ 0.040 & 0.060 & 0.800 & 0.060 & 0.040 \\ 0.000 & 0.040 & 0.060 & 0.840 & 0.060 \\ 0.000 & 0.000 & 0.040 & 0.080 & 0.880 \end{bmatrix};$$

- Low persistent blocks (Low π_{uu}):

$$\boldsymbol{\Pi} = \begin{bmatrix} 0.770 & 0.230 \\ 0.230 & 0.770 \end{bmatrix}, \quad \boldsymbol{\Pi} = \begin{bmatrix} 0.720 & 0.220 & 0.060 \\ 0.190 & 0.620 & 0.190 \\ 0.060 & 0.220 & 0.720 \end{bmatrix},$$

$$\boldsymbol{\Pi} = \begin{bmatrix} 0.700 & 0.200 & 0.080 & 0.020 & 0.000 \\ 0.150 & 0.650 & 0.150 & 0.040 & 0.010 \\ 0.050 & 0.150 & 0.600 & 0.150 & 0.050 \\ 0.010 & 0.040 & 0.150 & 0.650 & 0.150 \\ 0.000 & 0.020 & 0.080 & 0.200 & 0.700 \end{bmatrix};$$

- High intra-group connectivity ($\beta_{uu} > \beta_{uv}$):

$$\boldsymbol{B} = \begin{bmatrix} 0.332 & 0.030 \\ 0.030 & 0.431 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0.332 & 0.030 & 0.030 \\ 0.030 & 0.431 & 0.030 \\ 0.030 & 0.030 & 0.229 \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} 0.332 & 0.040 & 0.030 & 0.030 & 0.030 \\ 0.040 & 0.431 & 0.040 & 0.030 & 0.030 \\ 0.030 & 0.040 & 0.229 & 0.040 & 0.030 \\ 0.030 & 0.030 & 0.040 & 0.431 & 0.040 \\ 0.030 & 0.030 & 0.030 & 0.040 & 0.332 \end{bmatrix};$$

- High inter-group connectivity ($\beta_{uu} < \beta_{uv}$):

$$\boldsymbol{B} = \begin{bmatrix} 0.033 & 0.300 \\ 0.300 & 0.043 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} 0.033 & 0.300 & 0.300 \\ 0.300 & 0.043 & 0.300 \\ 0.300 & 0.300 & 0.023 \end{bmatrix},$$

$$\boldsymbol{B} = \begin{bmatrix} 0.033 & 0.300 & 0.200 & 0.100 & 0.100 \\ 0.300 & 0.043 & 0.300 & 0.200 & 0.100 \\ 0.200 & 0.300 & 0.023 & 0.300 & 0.200 \\ 0.100 & 0.200 & 0.300 & 0.043 & 0.200 \\ 0.100 & 0.100 & 0.200 & 0.300 & 0.033 \end{bmatrix}.$$

B Comparison of the initialization strategies in terms of computational time

In this section, we evaluate the computational time required by the standard VEM algorithm to converge, comparing different initialization strategies. All the simulations are performed on a *Standard_D8 d_v5* Linux virtual machine with 8 cores and 32 GB of RAM. Fig. 3 presents the computational time, measured in seconds, for the six initialization strategies, based on 50 samples for each of the 64 simulated scenarios. The DynSC initialization yields the fastest execution of the VEM algorithm, although its performance outcomes are generally unsatisfactory, as outlined in Sect. 4.1. The three initializations that demonstrate the best performance in terms of Adjusted Rand Index (ARI) and convergence to the global maximum - namely, k-means, BinSBM, and WgtSBM - exhibit similar and relatively low computational times, with a slight advantage for the two methods based on the static SB model. In contrast, random initialization and k-medoids show significantly higher computational times.

In addition to the execution times of the VEM algorithm with different initialization strategies, it is also important to evaluate the computational times associated with the initialization approaches themselves. In this regard, k-means is undoubtedly faster than the two initialization methods based on the estimation of a static SB model, primarily due to its simpler optimization process and the absence of probabilistic inference.

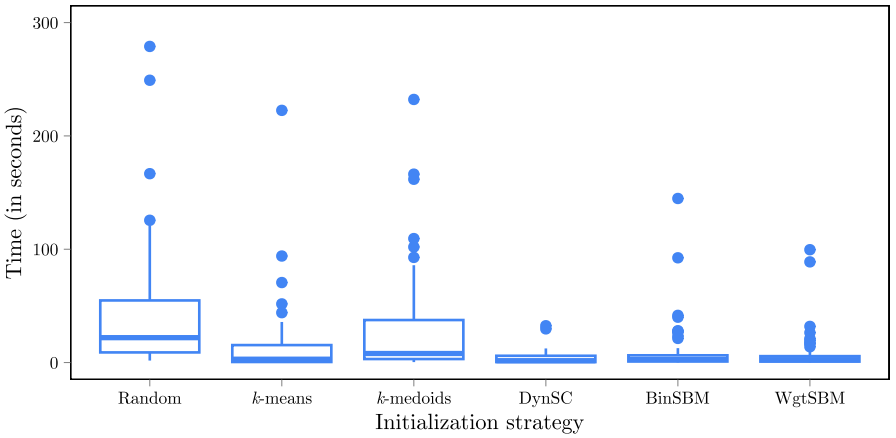


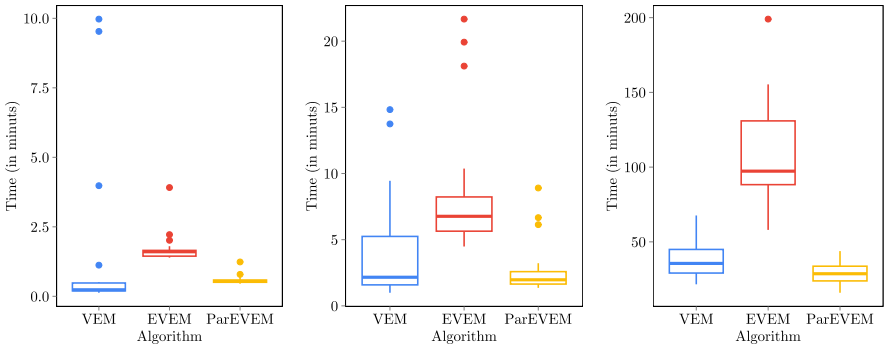
Fig. 3 Computational time for the standard VEM algorithm across six different initialization strategies, estimating 50 samples for each of the 64 simulated scenarios

Another important aspect to consider is scalability. The k-means algorithm is generally scalable; however, it can experience high computational complexity when applied to high-dimensional data. Improvements, like mini-batch k-means (Sculley, 2010), have been proposed to address these limitations. In contrast, the k-medoids algorithm has higher computational complexity and is less scalable. Spectral clustering also encounters scalability issues due to the spectral decomposition of the Laplacian matrix, though approximate methods like the Nyström approximation and sparse similarity method can enhance efficiency. Similarly, the initialization strategies based on estimating a static SB model lack scalability when applied to large network data. In conclusion, from a computational time perspective, k-means, BinSBM, and WgtSBM perform similarly for small networks. For large networks, k-means is the preferred choice, particularly when appropriately modified to address high dimensionality.

C Parallel implementation of the EVEM algorithm

The EVEM algorithm is considerably (and inherently) slower than the VEM algorithm due to its iterative procedure and the inclusion of evolutionary operators. A more realistic and informative comparison is between the computational time of the EVEM algorithm and the total computational time of multiple runs of the VEM. This approach is justified by two reasons: (*i*) it aligns with the procedure in Sect. 4.2, where the EVEM algorithm is compared in terms of ARI with the best outcome from multiple VEM executions; (*ii*) it accounts for the inherent difficulty in identifying a single initialization that consistently performs best across all samples and scenarios, requiring the use of multiple initializations when employing the VEM algorithm.

To address the computational demands of the EVEM algorithm, we develop a parallelized implementation. The parallelization specifically focuses on the two "Update" steps (Steps 1 and 3 in the procedure described in Sect. 3.1). In more details, the



(a) $n = 50,\ T = 10,\ k = 2,$ π_{uu} *low,* $\beta_{uu} > \beta_{uv}$

(b) $n = 50,\ T = 10,\ k = 3,$ π_{uu} *low,* $\beta_{uu} > \beta_{uv}$

(c) $n = 50,\ T = 10,\ k = 5,$ π_{uu} *low,* $\beta_{uu} > \beta_{uv}$

Fig. 4 Computational time for the standard VEM, the un-parallelized EVEM, and the parallelized EVEM algorithms for three simulated scenarios reported in Appendix A

candidate solutions within populations P_0 (Step 1) and P_2 (Step 3) are distributed across multiple threads; each thread handles the update process for a single candidate solution independently. This parallelization provides two key advantages on the estimation process. First, given a population of candidate solutions, it significantly reduces the execution time for the update steps. Second, it allows us to include an increased number of candidate solutions in populations P_0 and P_2, facilitating a more comprehensive exploration of the parameter space. This broader search accelerates the convergence of the algorithm toward the global maximum.

Results from a simulation study conducted on a *Standard_D8 d_v5* Linux virtual machine (8 cores, 32 GB RAM) demonstrate that the parallelized implementation is at least three times faster compared to the non-parallelized version, with greater improvements in computationally demanding scenarios, such as those involving larger numbers of nodes or latent blocks. Figure 4 shows the results of some comparisons of times between the standard VEM, the non-parallelized EVEM, and the parallelized EVEM algorithms. The simulations have been carried out considering three simulated scenarios characterized by the following settings: $n = 50$, $T = 10$, $k \in \{2, 3, 5\}$, low persistent blocks (low π_{uu}), and high intra-group connectivity ($\beta_{uu} > \beta_{uv}$). In particular, for the standard VEM algorithm, we consider the total computational time over 10 runs, considering the following initializations: k-means, BinSBM, WgtSBM, and 7 runs of random initializations. These results underline the efficiency and scalability of the proposed parallelized EVEM algorithm.

D Additional simulation results

In this section, we provide additional simulation results. Table 5 shows a comparison of the six proposed initialization strategies for the VEM algorithm in terms of ARI. For each scenario, we report only the strategy that yields the best result, indicated by the

Table 5 Average ARI between simulated and estimated clusters over 100 samples for the initialization strategy of the VEM algorithm that provides the best result (highest value of the log-likelihood function at convergence) across the 64 scenarios outlined in Appendix A: (*a*) *k*-means, (*b*) WgtSBM and (*c*) BinSBM. The columns labeled "High" and "Low" indicate cases with high and low persistence blocks in the transition matrix, respectively

			$\beta_{uu} > \beta_{uv}$		$\beta_{uu} < \beta_{uv}$	
			π_{uu}			
			High	Low	High	Low
$k = 2$	$n = 20$	$T = 6$	(*a*) 0.956	(*a*) 0.726	(*a*) 0.889	(*a*) 0.474
		$T = 10$	(*a*) 0.960	(*a*) 0.471	(*a*) 0.845	(*a*) 0.344
	$n = 50$	$T = 6$	(*a*) 1.000	(*b*) 0.993	(*a*) 0.993	(*b*) 0.929
		$T = 10$	(*a*) 1.000	(*a*) 0.843	(*a*) 0.995	(*a*) 0.750
	$n = 100$	$T = 6$	(*a*) 1.000	(*a*) 1.000	(*a*) 1.000	(*a*) 0.994
		$T = 10$	(*a*) 1.000	(*a*) 0.921	(*a*) 1.000	(*a*) 0.886
$k = 3$	$n = 20$	$T = 6$	(*b*) 0.664	(*a*) 0.226	(*b*) 0.566	(*b*) 0.148
		$T = 10$	(*b*) 0.677	(*a*) 0.213	(*b*) 0.578	(*a*) 0.088
	$n = 50$	$T = 6$	(*c*) 0.969	(*b*) 0.784	(*b*) 0.956	(*b*) 0.652
		$T = 10$	(*b*) 0.959	(*b*) 0.661	(*b*) 0.900	(*a*) 0.508
	$n = 100$	$T = 6$	(*b*) 0.969	(*c*) 0.958	(*a*) 0.969	(*b*) 0.875
		$T = 10$	(*b*) 0.930	(*b*) 0.864	(*a*) 0.958	(*a*) 0.739
$k = 5$	$n = 50$	$T = 6$	(*b*) 0.830	(*b*) 0.584	(*b*) 0.679	(*a*) 0.155
		$T = 10$	(*b*) 0.809	(*b*) 0.529	(*b*) 0.628	(*c*) 0.045
	$n = 100$	$T = 6$	(*b*) 0.912	(*b*) 0.835	(*b*) 0.965	(*c*) 0.738
		$T = 10$	(*b*) 0.875	(*c*) 0.757	(*b*) 0.942	(*a*) 0.608

highest log-likelihood value at convergence. Table 6 presents the sample proportion in which the log-likelihood value achieved through the EVEM algorithm surpasses the corresponding value obtained using the VEM algorithm.

Table 6 Sample proportion in which log-likelihood function at convergence is higher with the EVEM algorithm than with the VEM algorithm over 100 samples across the 64 scenarios outlined in Appendix A. The columns labeled "High" and "Low" indicate cases with high and low persistence blocks in the transition matrix, respectively

			$\beta_{uu} > \beta_{uv}$		$\beta_{uu} < \beta_{uv}$	
			π_{uu}			
			High	Low	High	Low
$k = 2$	$n = 20$	$T = 6$	0.170	0.370	0.530	0.900
		$T = 10$	0.180	0.610	0.630	0.940
	$n = 50$	$T = 6$	0.030	0.060	0.050	0.160
		$T = 10$	0.180	0.310	0.170	0.360
	$n = 100$	$T = 6$	0.180	0.290	0.330	0.380
		$T = 10$	0.740	0.840	0.580	0.610
$k = 3$	$n = 20$	$T = 6$	0.960	0.990	0.960	0.880
		$T = 10$	0.950	0.960	0.970	0.890
	$n = 50$	$T = 6$	0.680	0.840	0.800	0.950
		$T = 10$	0.740	0.900	0.860	0.960
	$n = 100$	$T = 6$	0.560	0.670	0.460	0.530
		$T = 10$	0.690	0.870	0.520	0.810
$k = 5$	$n = 50$	$T = 6$	1.000	1.000	0.980	0.920
		$T = 10$	0.960	1.000	0.920	0.960
	$n = 100$	$T = 6$	0.960	0.980	0.900	0.960
		$T = 10$	0.960	0.960	0.900	0.900

Acknowledgement L. Brusa thanks the financial support from the grant "Hidden Markov Models for Early Warning Systems" of the University of Milano-Bicocca (2022-ATEQC-0031). L. Brusa and F. Pennoni acknowledge the financial support from the grant "Hidden Markov Models for Early Warning Systems" of Ministero dell'Università e della Ricerca (PRIN 2022 TZEXKF) funded by the European Union - Next Generation EU, Mission 4, Component 2, CUP J53D23004990006.

Author contributions All authors wrote and reviewed the manuscript.

Funding Open access funding provided by Università degli Studi di Milano - Bicocca within the CRUI-CARE Agreement. Funding was provided by Università degli Studi di Milano-Bicocca (Grant Number 2022-ATEQC-0031), and by Ministero dell'Università e della Ricerca (Grant Number PRIN 2022 TZEXKF).

Data availability Data are available at the following website: https://www.sociopatterns.org/datasets/hospital-warddynamic-contact-network/.

Declarations

Conflict of interest The authors declare no Conflict of interest.

Ethical approval Any experimental research that is reported in the manuscript should have been performed with the approval of an appropriate ethics committee. Research carried out on humans must be in compliance with the Helsinki Declaration, and any experimental research on animals should follow internationally recognized guidelines. A statement to this effect must appear in the manuscript, including the name of the body which gave approval, with a reference number where appropriate. In this regard, please provide the missing information to comply with standard requirements.

Informed consent For this type of study formal consent is not required.

References

Andrews JL, McNicholas PD (2013) Using evolutionary algorithms for model-based clustering. Pattern Recognit Lett 34:987–992

Ashlock D (2004) Evolutionary computation for modeling and optimization. Springer, New York

Bäck T (1996) Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press, New York

Bartolucci F, Pandolfi S (2020) An exact algorithm for time-dependent variational inference for the dynamic stochastic block model. Pattern Recognit Lett 138:362–369

Bartolucci F, Pandolfi S, Pennoni F (2022) Discrete latent variable models. Annu Rev Stat Appl 6:1–31

Blei DM, Kucukelbir A, McAuliffe JD (2017) Variational inference: A review for statisticians. J Am Stat Assoc 112:859–877

Brusa L, Pennoni F, Bartolucci F (2024) Maximum likelihood estimation for discrete latent variable models via evolutionary algorithms. Stat Comput 34:1–15

Chiquet J, Donnet S, Barbillon P (2024). *sbm: Stochastic Blockmodels*

Daudin JJ, Picard F, Robin S (2008) A mixture model for random graphs. Stat Comput 18:173–183

Deb K (2001) Multi-Objective Optimization using Evolutionary Algorithms. John Wiley and Sons, Chichester

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm (with discussion). J. R. Stat Soc Series B Stat Methodol 39:1–38

Gao C, Ma Z, Zhang AY, Zhou HH (2017) Achieving optimal misclassification proportion in stochastic block models. J Mach Learn Res 18:1–45

Hruschka ER, Campello RJGB, Freitas AA, Ponce Leon F, de Carvalho AC (2009) A survey of evolutionary algorithms for clustering. IEEE Trans Syst Man Cybern 39:133–155

Hubert L, Arabie P (1985) Comparing partitions. J Classif 2:193–218

Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK (1999) An introduction to variational methods for graphical models. Mach Learn 37:183–233

Kaufman L, Rousseeuw PJ (1990) Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, Hoboken, New Jersey

Kullback S, Leibler R (1951) On information and sufficiency. Ann Math Stat 22:79–86

Liu F, Choi D, Xie L, Roeder K (2018). Global spectral clustering in dynamic networks. In: *Proceedings of the National Academy of Sciences*, pages 927–932. National Academy of Sciences

MacQueen J (1967). Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, Berkeley, CA, USA

Maechler M, Rousseeuw P, Struyf A, Hubert M, Hornik K (2023). *cluster: Cluster Analysis Basics and Extensions*

Matias C, Miele V (2017) Statistical clustering of temporal networks through a dynamic stochastic block model. J Royal Stat Soc Ser B 79:1119–1141

Matias C, Robin S (2014) Modeling heterogeneity in random graphs through latent space models: A selective review. ESAIM: Proc Surv 47:55–74

McNicholas SM, McNicholas PD, Ashlock DA (2021) An evolutionary algorithm with crossover and mutation for model-based clustering. J Classif 38:264–279

Nowicki K, Snijders TAB (2001) Estimation and prediction for stochastic blockstructures. J Am Stat Assoc 96:1077–1087

Olivella S, Pratt T, Imai K (2022) Dynamic stochastic blockmodel regression for network data: Application to international militarized conflicts. J Am Stat Assoc 117:1068–1081

Pernkopf F, Bouchaffra D (2005) Genetic-based EM algorithm for learning Gaussian mixture models. IEEE Trans Pattern Anal Mach Intell 27:1344–1348

R Core Team (2025) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria

Sculley D (2010). Web-scale *k*-means clustering. In: *Proceedings of the 19th International Conference on World Wide Web*, pages 1177–1178

Snijders TAB, Nowicki K (1997) Estimation and prediction for stochastic blockmodels for graphs with latent block structure. J Classif 14:75–100

Süveges M, Olhede SC (2023) Networks with correlated edge processes. J. R. Stat Soc Ser A Stat Soc 186:441–462

Vanhems P, Barrat A, Cattuto C, Pinton J-F, Khanafer N, Régis C, Kim B, Comte B, Voirin N (2013) Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. PloS One 8:1–9

Von Luxemburg U (2007) A tutorial on spectral clustering. Stat Comput 17:395–416

Wijayanto AW, Murata T (2019) Effective and scalable methods for graph protection strategies against epidemics on dynamic networks. Appl Netw Sci 4:1–31

Yang T, Chi Y, Zhu S, Gong Y, Jin R (2011) Detecting communities and their evolutions in dynamic social networks - a Bayesian approach. Mach Learn 82:157–189