



HAL
open science

A Deep Dynamic Latent Block Model for the Co-clustering of Zero-Inflated Data Matrices

Giulia Marchello, Marco Corneli, Charles Bouveyron

► **To cite this version:**

Giulia Marchello, Marco Corneli, Charles Bouveyron. A Deep Dynamic Latent Block Model for the Co-clustering of Zero-Inflated Data Matrices. 2022. hal-04150292v1

HAL Id: hal-04150292

<https://hal.science/hal-04150292v1>

Preprint submitted on 6 Oct 2022 (v1), last revised 4 Jul 2023 (v2)

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

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

A Deep Dynamic Latent Block Model for the Co-clustering of Zero-Inflated Data Matrices

Giulia Marchello

Université Côte d’Azur, Inria, CNRS, Laboratoire J.A.Dieudonné,
Maasai team, Nice, France.

and

Marco Corneli

Université Côte d’Azur, CNRS, Laboratoire CEPAM, Nice, France.

and

Charles Bouveyron

Université Côte d’Azur, Inria, CNRS, Laboratoire J.A.Dieudonné,
Maasai team, Nice, France.

October 5, 2022

Abstract

The simultaneous clustering of observations and features of data sets (known as co-clustering) has recently emerged as a central machine learning application to summarize massive data sets. However, most existing models focus on continuous and dense data in stationary scenarios, where cluster assignments do not evolve over time. This work introduces a novel latent block model for the dynamic co-clustering of data matrices with high sparsity. To properly model this type of data, we assume that the observations follow a time and block dependent mixture of zero-inflated distributions, thus combining stochastic processes with the time-varying sparsity modeling. To detect abrupt changes in the dynamics of both cluster memberships and data sparsity, the mixing and sparsity proportions are further modeled through systems of ordinary differential equations. The inference of the proposed model relies on an original variational EM procedure whose maximization step trains fully connected neural networks in order to solve the dynamical systems. The usual model selection question is also considered here through an original approach exploiting the ability of the model to work with empty clusters, thanks to the flexibility of deep neural networks. Numerical experiments on simulated and real world data sets demonstrate the effectiveness of the proposed methodology in the context of count data.

Keywords: Co-clustering, Latent Block Model, zero-inflated distributions, dynamic systems, VEM algorithm.

1 Introduction

1.1 Context

Along with the exponential increase in the massive availability of data in many domains, there is an increasing need for machine learning techniques able to summarize these flows of data. For instance, large social networking and online shopping platforms are eager to simultaneously clustering users and products to offer the best recommendations to their customers. Similarly, for pharmacovigilance, being an activity of monitoring adverse drug reaction (ADRs), it is very important to create consistent clusters of drugs and ADRs in order to be able to check for atypical phenomena in the development of claims. However, it often happens that classical clustering methods are not efficient in identifying homogeneous and interpretable groups when dealing with data sets containing a high number of variables (i.e. in high dimension). Co-clustering is particularly useful in the context of high dimensional data since it simultaneously clusters the rows (observations) and the columns (features) of a data matrix, thus providing useful summaries of the data. Moreover, in a wide range of application (e.g. processing, recommending systems, biomedical data, finance, genetics, etc.) there is a growing need to develop machine learning models to treat time-dependent data matrices, which can be also extremely sparse. Although the study in the field of co-clustering has been greatly expanded by many notable methods introduced in the last few decades, the development of dynamic co-clustering methods for sparse data still remains almost an unexplored territory.

1.2 Related Work

Related work in this context can be split into two main points.

Co-clustering and Latent Block Models Simultaneous clusters of observations and features can be of great help while analyzing a given data set. Co-clustering methods

can be distinguished into metric based approaches, such as non-negative matrix tri-factorization (NMTF, Labiod and Nadif, 2011; Ding et al., 2006), spectral co-clustering (Dhillon, 2001), information theory (Dhillon et al., 2003), and model-based co-clustering approaches (e.g. Bouveyron et al., 2019).

Among those approaches, model-based co-clustering is widely appreciated for its sound statistical foundations and its flexibility in terms of sparsity and data types. The cornerstone of model-based co-clustering is the popular latent block model (LBM) (Govaert and Nadif, 2003) that was initially introduced for the co-clustering of binary data matrices. LBM is based on the assumption that rows and columns are grouped in hidden clusters and that observations within a block (intersection of a row cluster and a column cluster) are independent and identically distributed. Whereas the original formulation of the model dealt with binary data only, the model has been extended in the last two decades to count data (Govaert and Nadif, 2010), continuous data (Lomet, 2012), categorical data (Keribin et al., 2015), ordinal data (Jacques and Biernacki, 2018; Corneli et al., 2020), functional data (Bouveyron et al., 2018), textual data (Bergé et al., 2019) and mixed-type data (Selosse et al., 2020). Recently, Boutalbi et al. (2020) also proposed the tensor latent block model (TLBM) for co-clustering, whose aim is to simultaneously cluster rows and columns of a 3D matrix, where covariates represent the third dimension. TLBM was also implemented for different types of data sets: continuous data (Gaussian TLBM), binary data (Bernoulli TLBM) and contingency tables (Poisson TLBM).

Dynamic models for clustering and co-clustering Whereas there is a decade-long literature about static model-based clustering and co-clustering methods, dynamic models are more recent in this context. It is worth noting that much more work has been made in the context of network clustering. In particular, for the Stochastic Block Model (SBM, Nowicki and Snijders, 2001) than for LBM, although SBM is a special case of LBM, not needing that data matrices are square and/or symmetrical. Yang et al. (2011) proposed

a dynamic version of SBM by allowing the cluster of each node to switch at time $t + 1$ depending on its current state at time t , in a Markovian framework, where the switching probabilities are collected into a transition matrix. In a more general framework, Matias and Miele (2017) showed that, in dynamic SBMs, it is not possible to let vary over time both the connectivity parameters and cluster memberships without incurring into identifiability issues. Recently, Marchello et al. (2022) proposed an extension of LBM allowing one to perform the simultaneous clustering of rows, columns and slices of a three dimensional counting tensor. Although being a first attempt to expand the LBM model to the dynamic case, this model has the limitation of not allowing cluster switches of rows/columns. In a different framework, Casa et al. (2021) prolong the latent block model to deal with longitudinal data, relying on the shape invariant model (Lindstrom, 1995) and Boutalbi et al. (2021) developed a model-based co-clustering method for sparse three-way data, where the third dimension can be seen as a discrete temporal one. Here, the sparsity is handled following the same assumption as in Ailem et al. (2017) that all blocks outside the main diagonal share the same parameter.

However, these methods, while a first step toward the dynamic expansion of co-clustering methods, can be used mostly for macroscopic analyses, as they do not allow for the temporal dependence of variables.

1.3 Contribution of this work

In this paper, we introduce a co-clustering method to deal with time evolving data matrices, potentially very sparse. In order to model the evolving generating process that gives the data and simultaneously accounts for the data sparsity, we assume that the observations follow a time and block dependent mixture of zero-inflated distributions. Since we aim at co-clustering the row and columns of the data matrices, we introduce two evolving latent random variables that model the group memberships of observations and features, respectively. Moreover, the parameters of the random variables and the data

sparsity proportion arise from three systems of ordinary differential equations that model the dynamics. Capturing the data dynamics is crucial in order to detect atypical phenomena that affected the generative process. For instance, if at a given time t_0 the value of some features suddenly increases for just one observation in a group, that observation will be very likely switched to another cluster, from t_0 on, when fitting our model to the data. This example suggests an interpretation of the results which is quite intuitive: a change in the affiliation of the observations/features to the clusters means that a change point has been detected, leaving space for further analysis to inspect the causes. Thus, we develop a highly interpretable co-clustering method allowing practitioners to obtain a faster visualization of the results in order to automate the data analysis. The proposed model can be used both as a tool for retrospective analysis, but also, and above all, as a tool for near real-time and analysis of the data progression.

1.4 Organization of the paper

This paper is organized as follows. Section 2 first recalls the original latent block model, before introducing the proposed generative model. In Section 3, the inference procedure is detailed. Section 4 presents various experiments on simulated data to test and evaluate the model performances. In Section 5, an application to a real world data set is presented. The London Bike sharing data set is analyzed in order to illustrate the potential of the proposed model. Section 6 provides some concluding remarks.

2 A Zero-Inflated dynamic LBM

In this section, we first recall the standard latent block model, then we introduce the Zero-Inflated Dynamic Latent Block model (Zero-Inflated dLBM).

Notation. The observed data are assumed to be collected into time evolving matrices, over the the interval $[0, T]$. We work in discrete time and assume that we have a time

partition of equally spaced points

$$0 = t_0 < t_1 < t_u \leq t_U = T.$$

Now up to rescaling, we can assume without loss of generality, that $t_{u+1} - t_u = 1$. Moreover, to simplify the exposition we omit the subscript u and, with a slight abuse of notation, we denote by t the generic time point t_u and by T the number of time points U . Thus, at (discretized) time t , we introduce the incidence matrix $X(t) \in \mathbb{N}^{N \times M}$ whose entry $X_{i,j}(t)$ represents its generic element and it contains the observation and features that took place between t and $t - 1$. The rows of $X(t)$ are indexed by $i = 1, \dots, N$ and the columns by $j = 1, \dots, M$.

2.1 The latent block model

Let us first recall the LBM model which consider a single data matrix. Assuming that the time period is restricted to one time point, the data is a $N \times M$ random matrix $X = \{X_{ij}\}_{i \in 1, \dots, N, j \in 1, \dots, M}$. In the LBM framework, rows and columns of X are assumed to be clustered respectively into Q and L groups, such that the data belonging to the same block are independent and identically distributed. More formally, the latent structure of rows and columns of X is identified by the following latent variables:

- $Z := \{z_{iq}\}_{i \in 1, \dots, N, q \in 1, \dots, Q}$ represents the clustering of rows into Q groups. Hence, row i belongs to cluster q if $z_{iq} = 1$;
- $W := \{w_{j\ell}\}_{j \in 1, \dots, M, \ell \in 1, \dots, L}$ represents the clustering of columns into L groups. Hence, column j belongs to cluster ℓ if $w_{j\ell} = 1$.

Moreover, Z and W are assumed to be independent and distributed according to multinomial distributions:

$$p(Z|\alpha) = \prod_{i=1}^N \prod_{q=1}^Q \alpha_q^{z_{iq}}, \quad p(W|\beta) = \prod_{j=1}^M \prod_{\ell=1}^L \beta_\ell^{w_{j\ell}},$$

where $\alpha_q = \mathbb{P}\{z_{iq} = 1\}$, $\beta_\ell = \mathbb{P}\{w_{j\ell} = 1\}$, $\sum_{q=1}^Q \alpha_q = 1$ and $\sum_{\ell=1}^L \beta_\ell = 1$. LBM further assumes that, conditionally to Z and W , the entries X_{ij} are independent and their distribution $\varphi(\cdot; \zeta)$ belongs to the same parametric family.:

$$X_{ij} \mid z_{iq}w_{j\ell} = 1 \sim \varphi(X_{ij}; \zeta_{q\ell}), \quad (1)$$

where ζ denotes the parameter set of the distribution $\varphi(X_{ij}, \cdot)$, which only depends on the given block. With these assumptions, the complete data likelihood can be written as:

$$\begin{aligned} p(X, Z, W \mid \alpha, \beta, \zeta) &= p(Z \mid \alpha)p(W \mid \beta)p(X \mid Z, W, \zeta) = \\ &= \prod_{i=1}^N \prod_{q=1}^Q \alpha_q^{z_{iq}} \prod_{j=1}^M \prod_{\ell=1}^L \beta_\ell^{w_{j\ell}} \prod_{i,q} \prod_{j,\ell} \varphi(X_{ij}; \zeta_{q\ell})^{z_{iq}w_{j\ell}}. \end{aligned} \quad (2)$$

Let us remind that the LBM model has been originally introduced by (Govaert and Nadif, 2003) for binary data matrices with the Bernoulli distribution.

2.2 A Zero-Inflated Dynamic Latent Block Model

We now aim at simultaneously clustering the rows and columns of the collection of data matrices $\{X(t)\}_t$ evolving along the time.

2.2.1 Clusters modeling

The rows (i.e. observations) and columns (i.e. features) of $X(t)$ are clustered into Q and L groups, respectively. Although Q and L are assumed fixed over time, each row/column is nevertheless allowed to change its cluster membership, in $[0, T]$. More formally, the cluster memberships of the rows and columns of X are identified by two evolving multinomial distributions, respectively parameterized by $\alpha(t)$ and $\beta(t)$:

$$Z(t) \sim \mathcal{M}(1, \alpha(t) := (\alpha_1(t), \dots, \alpha_Q(t))), \quad (3)$$

where $\mathcal{M}(1, \cdot)$ denotes the multinomial probability mass function and $\alpha_q(t) = \mathbb{P}\{z_{iq}(t) = 1\}$, with $\sum_{q=1}^Q \alpha_q(t) = 1$, at time $t = 0, \dots, T$.

Thus $Z(t) := \{z_{iq}(t)\}_{i \in 1, \dots, N; q \in 1, \dots, Q}$ represents the clustering of N rows into Q groups at a given time point t . In a similar fashion, for the column clusters, we assume:

$$W(t) \sim \mathcal{M}(1, \beta(t) := (\beta_1(t), \dots, \beta_L(t))), \quad (4)$$

where $\beta_\ell(t) = \mathbb{P}\{w_{j\ell}(t) = 1\}$ and $\sum_{\ell=1}^L \beta_\ell(t) = 1$.

The two random vectors Z and W are further assumed to be independent.

2.2.2 Sparsity Modeling

In order to model a potential extreme sparsity, the observed data are assumed to be modeled by a mixture of block-conditional Zero-Inflated (ZI) distributions, where the entries $X_{ij}(t)$ are conditionally independent:

$$X_{ij}(t) | Z_i(t), W_j(t) \sim ZI(\zeta_{Z_i(t), W_j(t)}, \pi(t)), \quad (5)$$

where ζ is a $Q \times L$ denotes the block-dependent parameter set of the distribution $\varphi(X_{ij}(t), \cdot)$, and $\pi(t)$ is a vector of length T that indicates the level of sparsity at any given time period. Being a mixture between a chosen distribution and a Dirac mass at zero, the Zero-Inflated distribution is used to account for a high sparsity in the data and can be formally written as:

$$\begin{cases} X_{ij}(t) | Z_i(t), W_j(t) = 0 & \text{with probability } \pi(t) \\ X_{ij}(t) | Z_i(t), W_j(t) \sim \varphi(X_{ij}, (t), \zeta_{Z_i(t), W_j(t)}) & \text{with probability } 1 - \pi(t) \end{cases} \quad (6)$$

Then, to model time-evolving sparsity, we rewrite Eq. (6) by introducing a hidden random matrix, $A \in \{0, 1\}^{N \times M}$, where for all i and j :

$$A_{ij}(t) \sim \mathcal{B}(\pi(t)),$$

with $\mathcal{B}(p)$ denoting the Bernoulli probability mass function of parameter p and such that

$$\begin{aligned} A_{ij}(t) = 1 &\Rightarrow X_{ij}(t) | Z_i(t), W_j(t) = 0 \\ A_{ij}(t) = 0 &\Rightarrow X_{ij}(t) | Z_i(t), W_j(t) \sim \varphi(X_{ij}, (t), \zeta_{Z_i(t), W_j(t)}). \end{aligned} \quad (7)$$

Among the possible distributions $\varphi(\cdot)$ that could be considered in this framework, we can cite the zero-inflated versions of the log-normal and the Gamma distributions for continuous data, or the zero-inflated Poisson (ZIP) distribution (Lambert, 1992) for count data.

2.2.3 Modeling the temporal evolution of the parameters

The last assumption concerns the modeling of clusters proportions and sparsity over time. In fact, the mixing parameters $\alpha(t)$ and $\beta(t)$ as well as the sparsity proportions $\pi(t)$ are assumed to be driven by a system of ordinary differential equations (ODEs). In this way, we are able to capture the temporal evolution of both the clusters composition and the (excess of) sparsity. In continuous time, the three dynamic systems reads as:

$$\frac{d}{dt}a(t) = f_Z(a(t)), \quad (8)$$

$$\frac{d}{dt}b(t) = f_W(b(t)), \quad (9)$$

$$\frac{d}{dt}c(t) = f_A(c(t)), \quad (10)$$

where $t \in [0, T]$, $f_Z : \mathbb{R}^Q \rightarrow \mathbb{R}^Q$, $f_W : \mathbb{R}^L \rightarrow \mathbb{R}^L$ and $f_A : \mathbb{R} \rightarrow \mathbb{R}$ are three unknown continuous functions $a : [0, T] \rightarrow \mathbb{R}^Q$, $b : [0, T] \rightarrow \mathbb{R}^L$ and $c : [0, T] \rightarrow \mathbb{R}$ are three continuously differentiable functions such that

$$\alpha_q(t) = \text{softmax}(a_q(t)) = \frac{e^{a_q(t)}}{\sum_{q=1}^Q e^{a_q(t)}}, \quad (11)$$

$$\beta_\ell(t) = \text{softmax}(b_\ell(t)) = \frac{e^{b_\ell(t)}}{\sum_{\ell=1}^L e^{b_\ell(t)}}, \quad (12)$$

$$\pi(t) = \frac{e^{c(t)}}{1 + e^{c(t)}}. \quad (13)$$

Then, since (as stated at beginning of Section 2) we work with discrete time points, the above dynamic systems reduce to their Euler schemes. For instance, Eq. (8) reduces to

$$a(t+1) = a(t) + f_Z(a(t)).$$

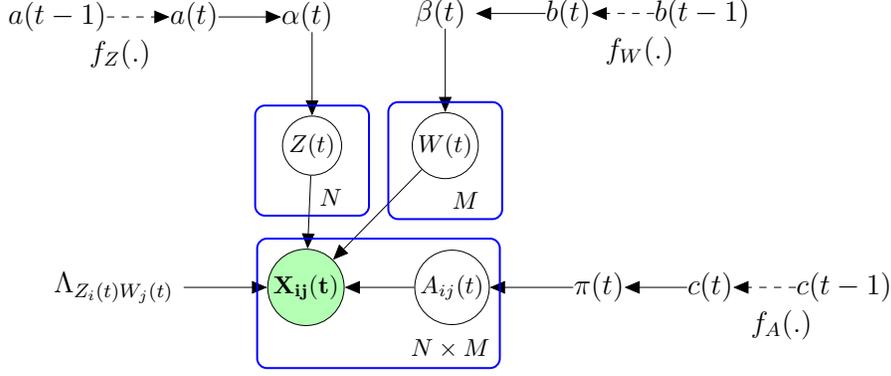


Figure 1: Graphical representation of the Zero-Inflated dLBM model.

A graphical representation of the model described so far, and named Zero-Inflated dLBM, can be seen in Figure 1.

2.3 The joint distribution

The model described so far can be adapted to any zero-inflated distribution. The first formulation as well as the most well-known concerns the Zero-Inflated Poisson, from an article by Lambert (1992). However, other distributions such as Zero-Inflated Negative Binomial (Ridout et al., 2001), Zero-Inflated Beta (Ospina and Ferrari, 2012), Zero-Inflated log-normal (Li et al., 2011) could be coupled with the present modeling.

In the following to ease the readability of the inference procedure we make use of the Zero-Inflated Poisson (ZIP) formulation to illustrate our approach. Hence, we can write $X_{ij}(t)|Z_i(t), W_j(t) \sim ZIP(\Lambda_{Z_i(t), W_j(t)}, \pi(t))$ and develop Eq. 6 as follows:

$$\begin{cases} X_{ij}(t)|Z_i(t), W_j(t) = 0 & \text{with probability } \pi(t) \\ X_{ij}(t)|Z_i(t), W_j(t) \sim \mathcal{P}(\Lambda_{Z_i(t), W_j(t)}) & \text{with probability } 1 - \pi(t) \end{cases} \quad (14)$$

where Λ is a $Q \times L$ matrix, denoting the block-dependent Poisson intensity function and $\pi(t)$ represents the sparsity at any given time period, with $t = 0, \dots, T$.

The model described so far has a set of parameters denoted by $\theta = (\Lambda, a(t), b(t), p(t))$

and a set of latent variables: $\{Z(t), W(t), A(t)\}$. We are now able to compute the likelihood of the complete data:

$$p(X, Z, W, A|\theta) = p(X|Z, W, A, \Lambda, \pi)p(A | \pi)p(Z|\alpha)p(W|\beta), \quad (15)$$

where:

$$p(X|A, Z, W, \Lambda, \pi) = \prod_{i=1}^N \prod_{j=1}^M \prod_{t=1}^T \mathbf{1}_{\{X_{ij}(t)=0\}}^{A_{ij}(t)} \left\{ \left(\frac{\Lambda_{Z_i(t)W_j(t)}^{X_{ij}(t)}}{X_{ij}(t)!} \exp(-\Lambda_{Z_i(t)W_j(t)}) \right)^{(1-A_{ij}(t))} \right\}, \quad (16)$$

$$p(A|\pi) = \prod_{i=1}^N \prod_{j=1}^M \prod_{t=1}^T \pi(t)^{A_{ij}(t)} (1 - \pi(t))^{(1-A_{ij}(t))}, \quad (17)$$

$$p(Z|\alpha) = \prod_{i=1}^N \prod_{q=1}^Q \prod_{t=1}^T \alpha_q(t)^{Z_{iq}(t)}, \quad (18)$$

$$p(W|\beta) = \prod_{j=1}^M \prod_{\ell=1}^L \prod_{t=1}^T \beta_{\ell}(t)^{W_{j\ell}(t)}. \quad (19)$$

3 The inference algorithm

Regarding the parameter estimation, the traditional procedure for such a model would be to maximize the log-likelihood $p(X|\theta)$. However, in our case, neither the direct maximization nor the classical EM algorithm (Dempster et al., 1977; Bishop, 2006) are feasible because of the impossibility to compute the joint conditional distribution of the latent variables, $p(Z, W|X, \theta)$, due to their interdependent double missing structure. An additional difficulty in parameter estimation is also the link that $\alpha(t)$, $\beta(t)$ and $\pi(t)$ have with their respective systems of differential equations, which do not allow the formulation of a closed updating formula. This is why, we rely on the Variational-EM algorithm combined with a Stochastic Gradient Descent (SGD) optimization step for the ODE part. Variational inference is usually applied to complex models involving missing values or relying on

latent structures (Jaakkola and Jordan, 1997; Jordan et al., 1998). The VEM algorithm has been shown to make relevant estimates of mixture models in different configurations (Govaert and Nadif, 2008).

3.1 Variational Assumption

Since we cannot compute the joint conditional distribution, $p(A, Z, W|X, \theta)$, we rely on a variational procedure which optimizes a lower bound of the likelihood. Let us thus introduce a variational distribution $q(\cdot)$ in order to decompose the likelihood as follows:

$$\log p(X|\theta) = \mathcal{L}(q; \theta) + KL(q(\cdot)||p(\cdot|X, \theta)), \quad (20)$$

where \mathcal{L} denotes a lower bound and is defined as:

$$\begin{aligned} \mathcal{L}(q, \theta) &= \sum_Z \sum_W \sum_A q(Z, W, A) \log \frac{p(X, A, Z, W|\theta)}{q(Z, W, A)} \\ &= E_{q(A, Z, W)} \left[\log \frac{p(X, A, Z, W|\theta)}{q(A, Z, W)} \right] \\ &= E_{q(A, Z, W)} [\log(p(X, A, Z, W|\theta))] - E_{q(A, Z, W)} [\log(q(A, Z, W))]. \end{aligned} \quad (21)$$

and KL indicates the Kullback-Liebler divergence between the true and the approximate posterior:

$$KL(q(\cdot)||p(\cdot|X, \theta)) = - \sum_A \sum_Z \sum_W q(A, Z, W) \log \frac{p(A, Z, W|X, \theta)}{q(A, Z, W)}. \quad (22)$$

Now, the objective is to find a distribution $q(\cdot)$ that maximizes the lower bound $\mathcal{L}(q, \theta)$. In order to allow the optimization of $\mathcal{L}(q, \theta)$, we further assume that $q(A, Z, W)$ factorizes as follows:

$$q(Z, W, A) = q(A)q(Z)q(W) = \prod_{i=1}^N \prod_{j=1}^M \prod_{t=1}^T q(A_{ij}(t)) \prod_{i=1}^N \prod_{t=1}^T q(Z_i(t)) \prod_{j=1}^M \prod_{t=1}^T q(W_j(t)). \quad (23)$$

3.2 VE-Step

The VE-step of the VEM algorithm aims at maximizing the lower bound in Eq. (21) with respect to the variational distribution $q(\cdot)$ while keeping θ fixed. Following Bishop (2006),

we derive the update equations for the factors $q(A)$, $q(Z)$, and $q(W)$, such that the log of the optimized factors are given by:

$$\log q^*(A) = E_{W,Z}[\log p(X, A, Z, W | \theta)], \quad (24)$$

$$\log q^*(Z) = E_{A,W}[\log p(X, A, Z, W | \theta)], \quad (25)$$

$$\log q^*(W) = E_{A,Z}[\log p(X, A, Z, W | \theta)]. \quad (26)$$

3.2.1 Optimization of the factor $q(A)$

Let us consider the derivation of the update equation for the factor $q(A)$. The sequential update for the factor $q(A)$ can be computed through the log of the optimized factor, where all the terms that do not depend on A are absorbed in the constant term.

Proposition 1. *Denoting by $\delta_{ij}(t) := q(A_{ij}(t) = 1)$ the variational success probability for $A_{ij}(t)$, the optimal update of $q(A)$ is:*

$$\delta_{ij}(t) = \frac{\exp(R_{ij}(t))}{1 + \exp(R_{ij}(t))}, \quad (27)$$

with:

$$\begin{aligned} R_{ij}(t) = & \log(\pi(t) \mathbf{1}_{\{X_{ij}(t)=0\}}) + \sum_{q=1}^Q \sum_{\ell=1}^L \left[-E[Z_{iq}(t)]E[W_{j\ell}(t)]X_{ij}(t) \log \Lambda_{q\ell} + \right. \\ & \left. + E[Z_{iq}(t)]E[W_{j\ell}(t)]\Lambda_{q\ell} \right] + \log X_{ij}(t)! - \log(1 - \pi(t)). \end{aligned} \quad (28)$$

Proof. Starting from Eq. 24, we use the decomposition in Eq. 15, and then we substitute the conditional distributions on the right-hand side. We denote by *const* those terms in

the lower bound not depending on $A_{ij}(t)$.

$$\begin{aligned}
\log(q^*(A)) &= E_{W,Z}[\log(p(X, Z, W, A|\theta))] + \text{const}; \\
&= E_{W,Z}[\log(p(X|Z, W, A, \Lambda, \pi))] + E_{W,Z}[\log(A|\pi)] + \text{const} \\
&= \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \left\{ A_{ij}(t) [\log \mathbf{1}_{\{X_{ij}(t)=0\}}] + (1 - A_{ij}(t)) \left[\sum_{q=1}^Q \sum_{\ell=1}^L [E[Z_{iq}(t)]E[W_{j\ell}(t)]X_{ij}(t) \log \Lambda_{q\ell} \right. \right. \\
&\quad \left. \left. - E[Z_{iq}(t)]E[W_{j\ell}(t)]\Lambda_{q\ell}] - \log X_{ij}(t)! \right] + A_{ij}(t) \log \pi(t) + (1 - A_{ij}(t)) \log(1 - \pi(t)) \right\} + \text{const} \\
&= \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \left\{ A_{ij}(t) \log \pi \mathbf{1}_{\{X_{ij}(t)=0\}} + (1 - A_{ij}(t)) \left[\sum_{q=1}^Q \sum_{\ell=1}^L [E[Z_{iq}(t)]E[W_{j\ell}(t)]X_{ij}(t) \log \Lambda_{q\ell} \right. \right. \\
&\quad \left. \left. - E[Z_{iq}(t)]E[W_{j\ell}(t)]\Lambda_{q\ell}] - \log X_{ij}(t)! + \log(1 - \pi(t)) \right] \right\} + \text{const} \\
&= \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \left\{ A_{ij}(t) \log \pi \mathbf{1}_{\{X_{ij}(t)=0\}} + A_{ij}(t) \left[\sum_{q=1}^Q \sum_{\ell=1}^L [-E[Z_{iq}(t)]E[W_{j\ell}(t)]X_{ij}(t) \log \Lambda_{q\ell} \right. \right. \\
&\quad \left. \left. + E[Z_{iq}(t)]E[W_{j\ell}(t)]\Lambda_{q\ell}] + \log X_{ij}(t)! - \log(1 - \pi(t)) \right] \right\} + \text{const} \\
&= \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \left\{ A_{ij}(t) \left[\log \pi \mathbf{1}_{\{X_{ij}(t)=0\}} + \sum_{q=1}^Q \sum_{\ell=1}^L [-E[Z_{iq}(t)]E[W_{j\ell}(t)]X_{ij}(t) \log \Lambda_{q\ell} \right. \right. \\
&\quad \left. \left. + E[Z_{iq}(t)]E[W_{j\ell}(t)]\Lambda_{q\ell}] + \log X_{ij}(t)! - \log(1 - \pi(t)) \right] \right\} + \text{const}.
\end{aligned} \tag{29}$$

We can then recognize the functional form of the Bernoulli distribution by indicating:

$$\begin{aligned}
\log q^*(A) &\propto \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T A_{ij}(t) \log \delta_{ij}(t) + (1 - A_{ij}(t)) \log(1 - \delta_{ij}(t)), \\
&\propto \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T A_{ij}(t) \frac{\log \delta_{ij}(t)}{1 - \log \delta_{ij}(t)}
\end{aligned} \tag{30}$$

where $\delta_{ij}(t)$ is defined as:

$$\delta_{ij}(t) = \frac{\exp(R_{ij}(t))}{1 + \exp(R_{ij}(t))}, \tag{31}$$

with $R_{ij}(t)$ defined as in Eq.(28).

□

3.2.2 Optimization of the factor $q(\mathbf{Z})$

Let us now consider the derivation of the update equation for the factor $q(Z)$. The sequential update for the factor $q(Z)$ can be computed through the logarithm of the optimized factor, where all the terms that do not depend on Z are absorbed in the constant term.

Proposition 2. Denoting by $\tau_{iq}(t) := q(Z_i q(t) = 1)$ the variational probability for $Z_{iq}(t)$, the optimal update of $q(Z)$ is:

$$\tau_{iq}(t) = \frac{r_{iq}(t)}{\sum_{q_0=1}^Q r_{iq_0}(t)}, \quad (32)$$

with $r_{iq}(t)$ is denoted by:

$$r_{iq}(t) \propto \exp \left(\sum_{j=1}^M \sum_{\ell=1}^L \left\{ (1 - E[A_{ij}(t)]) \left[E[W_{j\ell}(t)] X_{ij}(t) \log(\Lambda_{q\ell}) - E[W_{j\ell}(t)] \Lambda_{q\ell} \right] \right\} + \log(\alpha_q(t)) \right). \quad (33)$$

Proof. Starting from Eq. 25, we use the decomposition in Eq. 15, and then we substitute the conditional distributions on the right-hand side. We denote by *const* those terms in

the lower bound not depending on $Z_{iq}(t)$.

$$\begin{aligned}
\log q^*(Z|\theta) &= E_{W,A}[\log p(X, A, Z, W | \theta)] \\
&= E_{W,A}[\log(p(X | A, Z, W, \Lambda, \pi) + \log p(Z | \alpha))] \\
&= E_{W,A} \left[\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \left\{ (1 - A_{ij}(t)) \sum_{q=1}^Q \sum_{\ell=1}^L \left\{ Z_{iq}(t) W_{j\ell}(t) X_{ij}(t) \log(\Lambda_{q\ell}) - Z_{iq}(t) W_{j\ell}(t) \Lambda_{q\ell} \right\} \right. \right. \\
&\quad \left. \left. - (1 - A_{ij}(t)) \log(X_{ij}(t)!) \right\} \right] + \sum_{i=1}^N \sum_{q=1}^Q Z_{iq}(t) \log(\alpha_q(t)) + \text{const}, \\
&= \sum_{i=1}^N \sum_{j=1}^M (1 - E[A_{ij}(t)]) \left[\sum_{q=1}^Q \sum_{\ell=1}^L \left\{ Z_{iq}(t) E[W_{j\ell}(t)] X_{ij}(t) \log(\Lambda_{q\ell}) + \right. \right. \\
&\quad \left. \left. - Z_{iq}(t) E[W_{j\ell}(t)] \Lambda_{q\ell} \right\} \right] + \sum_{i=1}^N \sum_{q=1}^Q Z_{iq}(t) \log(\alpha_q(t)) + \text{const}, \\
&= \sum_{i=1}^N \sum_{q=1}^Q Z_{iq}(t) \left[\sum_{j=1}^M \sum_{\ell=1}^L \left\{ (1 - E[A_{ij}(t)]) \left[E[W_{j\ell}(t)] X_{ij}(t) \log(\Lambda_{q\ell}) + \right. \right. \right. \\
&\quad \left. \left. - E[W_{j\ell}(t)] \Lambda_{q\ell} \right] \right\} + \log(\alpha_q(t)) \right] + \text{const}.
\end{aligned} \tag{34}$$

We can then recognize the functional form of the multinomial distribution. Thus, we can write:

$$\log q^*(Z|\theta) = \sum_i \sum_t \sum_q Z_{iq}(t) \log r_{iq}(t) + \text{const}. \tag{35}$$

Taking the exponential on the two sides, we obtain:

$$q(Z_i) = \prod_{t=1}^T \prod_{q=1}^Q r_{iq}(t)^{Z_{iq}(t)}, \tag{36}$$

where $r_{iq}(t)$ is denoted by:

$$r_{iq}(t) \propto \exp \left(\sum_{j=1}^M \sum_{\ell=1}^L \left\{ (1 - E[A_{ij}(t)]) \left[E[W_{j\ell}(t)] X_{ij}(t) \log(\Lambda_{q\ell}) - E[W_{j\ell}(t)] \Lambda_{q\ell} \right] \right\} + \log(\alpha_q(t)) \right). \tag{37}$$

However, this distribution needs to be normalized because the matrix $Z(t)$ is a binary matrix and the elements sum to 1 over the values of Q . We can then obtain:

$$q(Z_i) = \prod_{t=1}^T \prod_{q=1}^Q \tau_{iq}(t)^{Z_{iq}(t)}, \quad (38)$$

where

$$\tau_{iq}(t) = \frac{r_{iq}(t)}{\sum_{q_0=1}^Q r_{iq_0}(t)}. \quad (39)$$

□

3.2.3 Optimization of the factor $q(W)$

Let us now consider the derivation of the update equation for the factor $q(W)$. The sequential update for the factor $q(W)$ can be computed through the log of the optimized factor, where all the terms that do not depend on W are absorbed in the constant term.

Proposition 3. *Similarly for the latent variable W , denoting by $\eta_{j\ell}(t) := q(W_j\ell(t) = 1)$ the variational probability for $W_{j\ell}(t)$, the optimal update of $q(W)$ is:*

$$\eta_{j\ell}(t) = \frac{s_{j\ell}(t)}{\sum_{\ell_o=1}^L s_{j\ell_o}(t)}, \quad (40)$$

where :

$$s_{j\ell}(t) \propto \exp \left(\sum_{i=1}^N \sum_{q=1}^Q \left\{ (1 - E[A_{ij}(t)]) \left[E[Z_{iq}(t)] X_{ij}(t) \log(\Lambda_{q\ell}) - E[Z_{iq}(t)] \Lambda_{q\ell} \right] \right\} + \log(\beta_\ell(t)) \right). \quad (41)$$

The proof is symmetric to the one developed for $\tau_{iq}(t)$ in Proposition 2.

3.3 Variational M-Step

In order to obtain the updating of the parameter set θ , the objective of the M-Step is the maximization of the lower bound $\mathcal{L}(q, \theta)$ with respect to $\theta = (\Lambda, \alpha(t), \beta(t), \pi(t))$, while holding the variational distribution $q(\cdot)$ fixed.

Proposition 4. By developing the Eq. 21, the variational lower bound $\mathcal{L}(q, \theta)$ can be written as:

$$\begin{aligned}
\mathcal{L}(q, \theta) = & \sum_{t=1}^T \sum_{i=1}^N \sum_{j=1}^M \left\{ \delta_{ij}(t) \log(\pi(t) \mathbf{1}_{\{X_{ij}(t)=0\}}) + (1 - \delta_{ij}(t)) \left[\log(1 - \pi(t)) + \right. \right. \\
& + \sum_{q=1}^Q \sum_{\ell=1}^L \left\{ \tau_{iq}(t) \eta_{j\ell}(t) X_{ij}(t) \log \Lambda_{q\ell} - \tau_{iq}(t) \eta_{j\ell}(t) \Lambda_{q\ell} \right\} \left. \right] + \\
& - (1 - \delta_{ij}(t)) \log(X_{ij}(t)!) \left. \right\} + \sum_{t=1}^T \sum_{i=1}^N \sum_{q=1}^Q \tau_{iq}(t) \log(\alpha_q(t)) + \\
& + \sum_{t=1}^T \sum_{j=1}^M \sum_{\ell=1}^L \eta_{j\ell}(t) \log(\beta_\ell(t)) - \sum_{t=1}^T \sum_{i=1}^N \sum_{q=1}^Q \tau_{iq}(t) \log \tau_{iq}(t) + \\
& - \sum_{t=1}^T \sum_{j=1}^M \sum_{\ell=1}^L \eta_{j\ell}(t) \log(\eta_{j\ell}(t)) - \sum_{t=1}^T \sum_{i=1}^N \sum_{j=1}^M \left(\delta_{ij}(t) \log(\delta_{ij}(t)) + (1 - \delta_{ij}(t)) \log(1 - \delta_{ij}(t)) \right).
\end{aligned} \tag{42}$$

Proof. Starting from Eq. 21 we obtain the final expression of the variational lower bound $\mathcal{L}(q, \theta)$ by developing the expression:

$$\begin{aligned}
\mathcal{L}(q, \theta) = & \sum_{A, Z, W} q(A, Z, W) \log \frac{p(X|A, Z, W, \Lambda) p(A | \pi) p(Z|\alpha) p(W|\beta)}{q(A, Z, W)} \\
= & E_{A, Z, W} \left[\log \frac{p(X|A, Z, W, \Lambda) p(A | \pi) p(Z|\alpha) p(W|\beta)}{\prod_{i=1}^N \prod_{j=1}^M \prod_{t=1}^T q(A_{ij}(t)) \prod_{i=1}^N \prod_{t=1}^T q(Z_i(t)) \prod_{j=1}^M \prod_{t=1}^T q(W_j(t))} \right] \\
= & E_{A, Z, W} [\log p(X|A, Z, W, \Lambda)] + E_A [\log p(A | \pi)] + E_Z [\log p(Z|\alpha)] + \\
& + E_W [\log p(W|\beta)] - E_Z [\log \prod_i q(Z_i)] + \\
& - E_W [\log \prod_j q(W_j)] - E_A [\log \prod_i \prod_j q(A_{ij})].
\end{aligned} \tag{43}$$

Then we substitute the results obtained in the VE-Step, denoting $E[A_{ij}(t)] = \delta_{ij}(t)$, $E[Z_{iq}(t)] = \tau_{iq}(t)$ and $E[W_{j\ell}(t)] = \eta_{j\ell}(t)$, in order to obtain the final expression of the

lower bound that can be written as follows:

$$\begin{aligned}
\mathcal{L}(q, \theta) = & \sum_{t=1}^T \sum_{i=1}^N \sum_{j=1}^M \left\{ \delta_{ij}(t) \log(\pi(t) \mathbf{1}_{\{X_{ij}(t)=0\}}) + (1 - \delta_{ij}(t)) \left[\log(1 - \pi(t)) + \right. \right. \\
& + \sum_{q=1}^Q \sum_{\ell=1}^L \left. \left\{ \tau_{iq}(t) \eta_{j\ell}(t) X_{ij}(t) \log \Lambda_{q\ell} - \tau_{iq}(t) \eta_{j\ell}(t) \Lambda_{q\ell} \right\} \right] + \\
& \left. - (1 - \delta_{ij}(t)) \log(X_{ij}(t)!) \right\} + \sum_{t=1}^T \sum_{i=1}^N \sum_{q=1}^Q \tau_{iq}(t) \log(\alpha_q(t)) + \\
& + \sum_{t=1}^T \sum_{j=1}^M \sum_{\ell=1}^L \eta_{j\ell}(t) \log(\beta_\ell(t)) - \sum_{t=1}^T \sum_{i=1}^N \sum_{q=1}^Q \tau_{iq}(t) \log \tau_{iq}(t) + \\
& - \sum_{t=1}^T \sum_{j=1}^M \sum_{\ell=1}^L \eta_{j\ell}(t) \log(\eta_{j\ell}(t)) - \sum_{t=1}^T \sum_{i=1}^N \sum_{j=1}^M \left(\delta_{ij}(t) \log(\delta_{ij}(t)) + (1 - \delta_{ij}(t)) \log(1 - \delta_{ij}(t)) \right).
\end{aligned} \tag{44}$$

□

3.3.1 Update of Λ

Here our goal is to derive the update of the Zero-inflated Poisson intensity parameter, Λ . The variational distribution $q(A, Z, W)$ is kept fixed, while the lower bound is maximized with respect to Λ , to obtain its update, $\hat{\Lambda}$. However, in case other zero-inflated distributions are chosen, this step must obviously be adapted to the new distribution, although the procedure of the derivation does not change.

Proposition 5. *The updating formula of Λ is obtained by maximizing $\mathcal{L}(q, \theta)$ with respect to the parameter and it can be written as follows:*

$$\hat{\Lambda}_{q\ell} = \frac{\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \tau_{iq}(t) \eta_{j\ell}(t) \left(X_{ij}(t) - \delta_{ij}(t) X_{ij}(t) \right)}{\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \tau_{iq}(t) \eta_{j\ell}(t) \left(1 - \delta_{ij}(t) \right)}. \tag{45}$$

Proof. To find the optimal update expression of Λ we compute the derivative of the lower bound $\mathcal{L}(q, \theta)$ in Eq. 42 with respect to Λ :

$$\frac{\partial \log \mathcal{L}(q, \theta)}{\partial \Lambda_{q\ell}} = \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T (1 - \delta_{ij}(t)) \left[\frac{\tau_{iq}(t) \eta_{j\ell}(t) X_{ij}(t)}{\Lambda_{q\ell}} - \tau_{iq}(t) \eta_{j\ell}(t) \right], \tag{46}$$

Equating $\frac{\partial \log \mathcal{L}(q, \theta)}{\partial \Lambda_{q\ell}}$ to zero leads to:

$$\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T (1 - \delta_{ij}(t)) \left[\tau_{iq}(t) \eta_{j\ell}(t) X_{ij}(t) - \tau_{iq}(t) \eta_{j\ell}(t) \Lambda_{q\ell} \right] = 0, \quad (47)$$

and then

$$\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T (1 - \delta_{ij}(t)) \tau_{iq}(t) \eta_{j\ell}(t) \Lambda_{q\ell} = \sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \tau_{iq}(t) \eta_{j\ell}(t) \left[X_{ij}(t) - X_{ij}(t) \delta_{ij}(t) \right]. \quad (48)$$

We finally get:

$$\hat{\Lambda}_{q\ell} = \frac{\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \tau_{iq}(t) \eta_{j\ell}(t) \left(X_{ij}(t) - \delta_{ij}(t) X_{ij}(t) \right)}{\sum_{i=1}^N \sum_{j=1}^M \sum_{t=1}^T \tau_{iq}(t) \eta_{j\ell}(t) \left(1 - \delta_{ij}(t) \right)}. \quad (49)$$

□

3.3.2 Update of $\alpha(t)$, $\beta(t)$ and $\pi(t)$ through deep neural networks

The mixture proportions, $\alpha(t)$ and $\beta(t)$, and the sparsity parameter, $\pi(t)$, are driven by three systems of differential equations, in Eqs. (8) (9) and (10), respectively. As we assumed that the functions f_A , f_W and f_Z are continuous, we propose here to parametrize them using three fully connected **neural networks** (Gent and Sheppard, 1992). Thus, optimizing the lower bound in Eq. (42) with respect to $\alpha(t)$, $\beta(t)$ and $\pi(t)$, reduces to maximize it with respect to the parameters of the neural networks, ω_A, ω_Z and ω_W , as well as to the initial values $a(0), b(0)$ and $c(0)$. In particular, we denote with $\omega(h)$ the set of weights of the neural network settled for the updating of the related parameter at iteration h . The initial set of weight is randomly sampled, $\omega(0) = \{\omega(0)\}_{k=1}^K$, and along the iterations they are updated, such that:

$$\omega_k(h) = \omega_k(h-1) - \gamma \nabla \mathcal{L}_{\omega_k(h)}. \quad (50)$$

where γ is the learning rate, in the experiments they are $\gamma_A, \gamma_Z, \gamma_W = 1e-4$. The maximization is implemented in PyTorch via automatic differentiation (Paszke et al.,

2017) and relies on stochastic optimisation (ADAM, Kingma and Ba, 2014). Thanks to back-propagation the updated networks provide us with estimates of $\alpha(t)$, $\beta(t)$ and $\pi(t)$. The inference procedure is summarized in Algorithm 1.

3.4 Initialization and model selection

When dealing with clustering methods based on the EM algorithm, the initialization and the selection of the appropriate numbers of clusters (for rows and columns here) are two issues which deserve an appropriate treatment. The issues related to these two points are slightly complicated here by the use of deep neural networks for modeling the dynamics of cluster and sparsity proportions. Despite this apparent difficulty due to the intrinsic complexity of these networks, they will nevertheless offer some unexpected flexibilities that we may use to lower the computational cost of the whole algorithm. Indeed, and as it will be illustrated in the following numerical experiments (Section 4), the use of deep neural networks for modeling the row and column cluster proportions will allow our algorithm to work with some empty clusters.

Therefore, in the objective of avoiding the usual computationally demanding procedure of testing all pairs of row and column cluster numbers, we propose the following strategy for both initialization and model selection.

- First, we select a single specific slice of the data $X_{t_{init}}$ and apply on it a static version of our Zip-dLBM algorithm for a list of pairs of cluster numbers, i.e. (q, ℓ) for $q = 2, \dots, Q_{max}$ and $\ell = 2, \dots, L_{max}$. We then use the ICL criterion (Integrated Completed Likelihood, Biernacki et al. (2000)) to select the most appropriate row and column cluster numbers for this specific slice of data. Let us remind that the ICL (Integrated Completed Likelihood) criterion aims at approximating the complete-

Algorithm 1 VEM-SGD Algorithm (for the Zero-Inflated Poisson distribution)

Require: $X, Q, L, max.iter, \alpha(t), \beta(t), \pi(t), \Lambda$ from Initialization.

Initialization of $\tau(t)$ and $\eta(t)$: sampling from $\mathcal{M}(\alpha(t))$ and $\mathcal{M}(\beta(t))$, respectively;

Initialization of $\delta(t)$: matrix of 1, then setting $\delta(t) = 0$ when $X > 0$;

for it = 1 to *max.iter* **do**

VE-Step:

for p = 1 to Fixed.Point **do**

Update $\delta(t), \tau(t), \eta(t)$:

$$\delta_{ij}(t) = \frac{\exp(R_{ij}(t))}{(1 + \exp(R_{ij}(t)))},$$

where:

$$R_{ij}(t) = \log(\pi(t)\mathbf{1}_{\{X_{ij}(t)=0\}}) + \sum_{q=1}^Q \sum_{\ell=1}^L \left[-\tau_{iq}(t)\eta_{j\ell}(t)X_{ij}(t) \log \Lambda_{q\ell} + \tau_{iq}(t)\eta_{j\ell}(t)\Lambda_{q\ell} \right] + \log X_{ij}(t)! - \log(1 - \pi(t)).$$

$$\tau_{iq}(t) = \frac{1}{D_q} \exp \left(\sum_{j=1}^M \sum_{\ell=1}^L \left\{ (1 - \delta_{ij}(t)) \left[\eta_{j\ell}(t)X_{ij}(t) \log(\Lambda_{q\ell}) - \eta_{j\ell}(t)\Lambda_{q\ell} \right] \right\} + \log(\alpha_q(t)) \right).$$

$$\eta_{j\ell}(t) = \frac{1}{D_\ell} \exp \left(\sum_{i=1}^N \sum_{q=1}^Q \left\{ (1 - \delta_{ij}(t)) \left[\tau_{iq}(t)X_{ij}(t) \log(\Lambda_{q\ell}) - \tau_{iq}(t)\Lambda_{q\ell} \right] \right\} + \log(\beta_\ell(t)) \right).$$

with D_q and D_ℓ normalizing constants.

end for

M-Step:

Update $\theta = (\Lambda, \pi(t), \alpha(t), \beta(t))$.

$$\hat{\Lambda}_{q\ell} = \frac{\sum_{i,j,t} \left\{ \tau_{iq}(t)\eta_{j\ell}(t) \left(X_{ij}(t) - \delta_{ij}(t)X_{ij}(t) \right) \right\}}{\sum_{i,j,t} \left\{ \tau_{iq}(t)\eta_{j\ell}(t) \left(1 - \delta_{ij}(t) \right) \right\}}.$$

for epoch = 1 to Epochs **do**

Update $\hat{\alpha}(t), \hat{\beta}(t), \hat{\pi}(t)$:

Loss Evaluation;

Algorithm backpropagation;

Numerical optimization with SGD.

end for

end for

data integrated log-likelihood and can be derived for the Zip-dLBM model as follows:

$$\begin{aligned}
 ICL(Q, L) = \log p(X, \hat{Z}, \hat{W}; \hat{\theta}) - \frac{Q-1}{2} \log N + \\
 - \frac{L-1}{2} \log M - \frac{QL}{2} \log(NM) - \frac{1}{2} \log(NM).
 \end{aligned} \tag{51}$$

The pair (\hat{Q}, \hat{L}) that leads to the highest value for the ICL is considered as the most meaningful cluster numbers for the considered slice of data $X_{t_{init}}$. Remark that, unless a further specific notice, the slice $X_{t_{init}}$ considered for this step in our experiments will be the first slice of the data, i.e. X_{t_0} .

- Second, in order to initialize our VEM-SGD algorithm (see Algorithm 1) with useful initial values for model parameters, we initiate a cascade process as follows in order to propagate the parameter estimates obtained on the slice $X_{t_{init}}$ to the following slices. Fixing for the moment the numbers of row and column clusters to (\hat{Q}, \hat{L}) , we run the static version of our Zip-dLBM algorithm on the next slice $X_{t_{init}+1}$ with the parameters $\hat{\theta}_{t_{init}}$ as initial values. Then, the estimated parameters $\hat{\theta}_{t_{init}+1}$ are used as initialization of the static Zip-dLBM on the slice $X_{t_{init}+2}$, and so on for the following slices. This strategy allows to provide initial values for all model parameters $\hat{\theta}(t)$, for $t = 1, \dots, T$.
- Finally, as we expect that the choice of \hat{Q} row and \hat{L} column cluster components could not be the best for all slices of the data set, the VEM-SGD algorithm (see Algorithm 1) will be then run with more components than considered in the initialization. Indeed, we run the VEM-SGD algorithm with $Q_{max} \geq \hat{Q}$ and $L_{max} \geq \hat{L}$ cluster components. Then, part of the model parameters are initialized with $\hat{\theta}(t)$ obtained via the initialization procedure described above (see Algorithm 2) and the remaining parameters, corresponding to the additional row and column clusters are set to zero. Thus, we aim at exploiting the potential "blessing" of the use of deep neural networks allowing our VEM-SGD algorithm to start with some empty clusters. These empty clusters will have the possibility to be activated later in

the inference process, if needed. Therefore, we avoid the usual computationally demanding procedure of running the whole algorithm with all pairs of row and column cluster numbers for the whole data set. This strategy allows our approach to scale to massive data sets in a reasonable computation time and with satisfying results, as it will be illustrated in the next section.

4 Numerical experiments

The main purpose of this section is to highlight the most important features of our zero-inflated dLBM algorithm over simulated data sets in the Poisson scenario. We aim at demonstrating the validity of the inference algorithm and model selection criterion presented in the previous sections. The first experiment consists in applying Zip-dLBM to a specific data set with evolving block pattern and sparsity to show that it recovers the data structure. The second experiment shows that Zip-dLBM is able to uncover clusters being initially empty, filling up over time, then emptying again. The third experiment shows the robustness of Zip-dLBM when the initial number of clusters is not the actual one, thus testing the performance of the model in case of poor initialization. The fourth experiment demonstrates the model selection procedure on 50 simulated data sets. In the fifth experiment, we finally compare the performances of Zip-dLBM with the dLBM model in two simulated scenarios, with constant and evolving sparsity, respectively. All the experiments on simulated data were realized on data sets with $N = 600$ rows, $M = 400$ columns and $T = 50$ time instants.

4.1 Introductory example

As a first example, we simulate a data set with dimension $600 \times 400 \times 50$ and with $Q = 3$ groups of rows, $L = 2$ groups of columns. The level of sparsity ranges from 80% to 90%

Algorithm 2 Initialization Algorithm

Step 1: Static model selection**Require:** $X, Q_{min}, Q_{max}, L_{min}, L_{max}, max_iter, n.sim.$ **for** $Q = Q_{min}$, to $Q = Q_{max}$ **do****for** $L = L_{min}$, to $L = L_{max}$ **do**Initialize randomly $\alpha, \beta, \pi, \Lambda$;Run a static version of Zip-dLBM on $t = 1$, computing the ICL;**end for****end for**Obtain Q^* and L^* that gives the highest ICL value.**Step 2: Cascade Process****Require:** $X, Q^*, L^*, max_iter.$ **for** $t = 1$ to T **do****if** $t = 1$ **then**Initialize randomly $\alpha, \beta, \pi, \Lambda$;Run a static version of Zip-dLBM on $t = 1$;Store the results $\alpha(t = 1), \beta(t = 1), \pi(t = 1), \Lambda.$ **else**Initialize $\alpha(t - 1), \beta(t - 1), \pi(t - 1), \Lambda$;Run a static version of Zip-dLBM on t ;Store the results $\alpha(t), \beta(t), \pi(t), \Lambda.$ **end if****end for**

in the time period. The values of the other simulated parameters in this experiment are:

Cluster	α	β
1	0.2 to 0.8	0.1 to 0.99
2	0.18 to 0.14	0.99 to 0.1
3	0.6 to 0.06	-

$$\Lambda = \begin{bmatrix} 6 & 4 \\ 1 & 2 \\ 7 & 3 \end{bmatrix}$$

We apply Zip-dLBM to the simulated data set with the actual values of Q and L to show the ability of the model to fully recover the model parameters.

Figure 2 shows the evolution of the the lower bound, expressed in Eq. 42, that Zip-dLBM aims to maximize. We can notice the convergence is reached in less than 10 iterations, in this example.

Figure 3, displays the reorganized incidence matrices at time instants $t = 10$ and $t = 30$, respectively: the rows and columns of the incidence matrix are permuted according to the estimates of the latent variables \hat{Z} and \hat{W} , in such a way that nearby rows (columns) belong to the same cluster of rows (columns). The blocks are also delimited by black dashed lines. The density of points in each block depends on the intensity function of the Poisson distribution Λ .

Figure 4 shows the evolution of the estimated mixture parameters $\alpha(\hat{t})$, $\beta(\hat{t})$ and $\pi(\hat{t})$ along the time period, represented on the x-axes. These parameters are estimated through the stochastic gradient descent technique, linked with the neural networks. By looking at these figures, we see the true parameters on the left column, the output of the initialization procedure in the middle and the results the Zip-dLBM estimates on the right. The comparison between the simulated and estimated parameter evolution shows that the model fully recovers the actual values over time, modulo the switched labels for the mixture proportions.

From these results we can clearly see that our algorithm perfectly identifies the composition of the original clusters and it recovers the evolution of the mixing proportion over time.

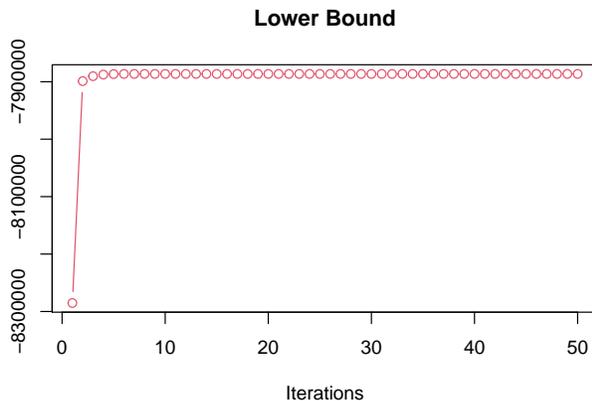


Figure 2: Lower bound maximization throughout the iterations of the Zip-dLBM algorithm.

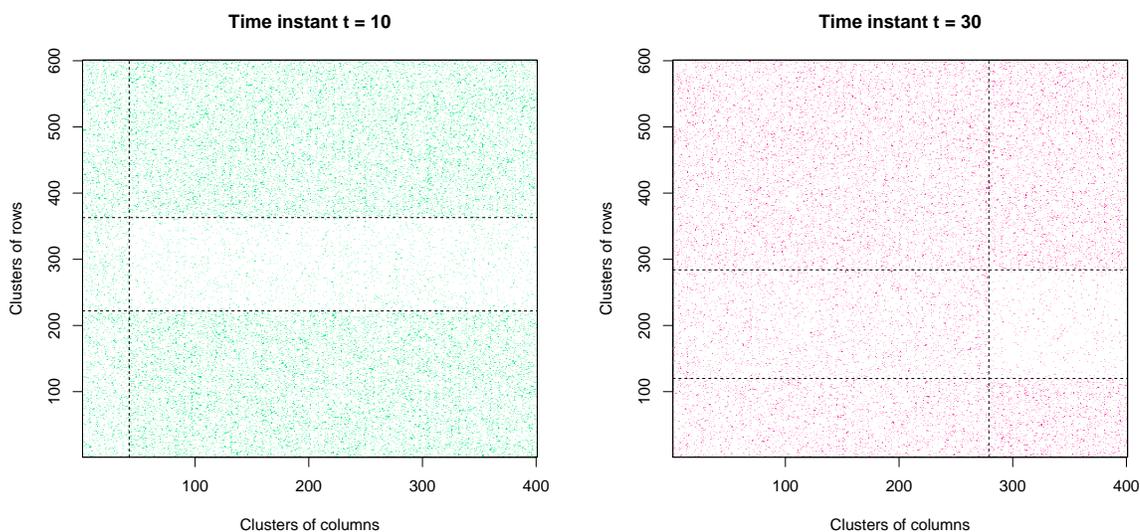
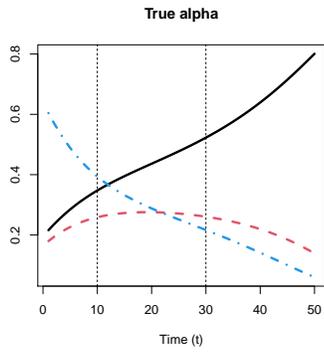


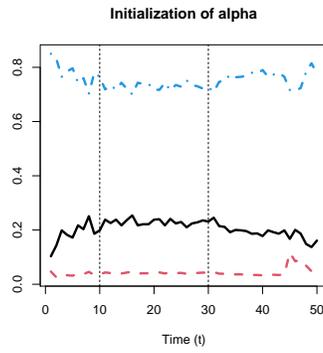
Figure 3: Reorganized incidence matrices at time instants $t = 10$ and $t = 30$ according to the estimates of the cluster memberships. Nearby rows (columns) belong to the same cluster of rows (columns). The blocks are also delimited by black dashed lines.

4.2 Robustness of the initialization procedure

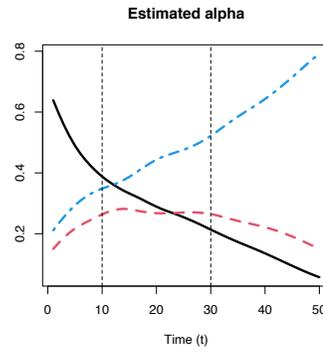
In this section we perform two experiments to test the robustness of the model to initialization. In the first experiment, we initialize parameters with wrong number of clusters,



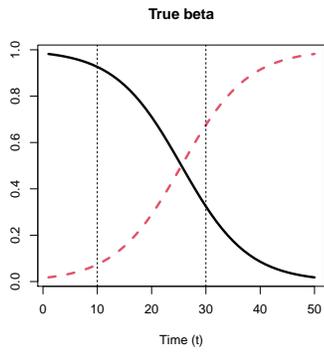
(a) True $\alpha(t)$.



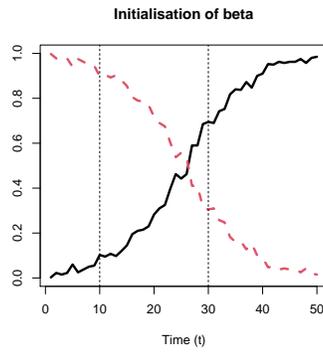
(b) Initialization of $\alpha(t)$.



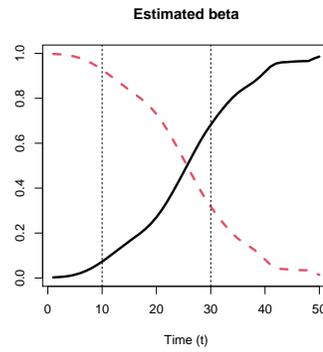
(c) Estimated $\alpha(t)$.



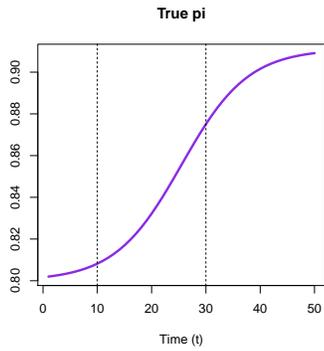
(d) True $\beta(t)$.



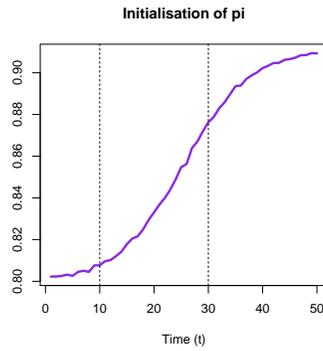
(e) Initialization of $\beta(t)$.



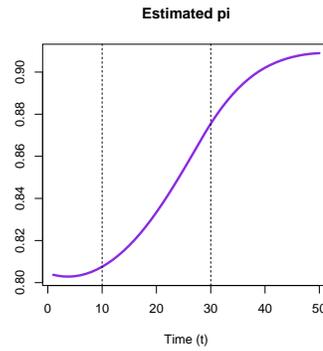
(f) Estimated $\beta(t)$.



(g) True $\pi(t)$.



(h) Initialization of $\pi(t)$.



(i) Estimated $\pi(t)$.

Figure 4: Evolution of the true (left), initialized (center) and estimated (right) proportions of the parameters $\alpha(t)$, $\beta(t)$ and $\pi(t)$, respectively.

while in the second experiment the data are simulated with particularly complex dynamics. In fact, we test the model’s ability to identify a cluster that is empty at the beginning of the period, which fills up and then empties again. As for the first experiment to test the robustness of our initialization strategy, Zip-dLBM was intentionally initialized with a higher than the optimal number of clusters. In fact, although the data were simulated with $Q = 3$ and $L = 2$, both the initialization process and Zip-dLBM were run with $Q = 5$ and $L = 4$. Figure 5 shows on the left column the evolution of the simulated mixture proportions and the sparsity parameter, in the middle column their initialization, and on the right column the results of the estimates provided by Zip-dLBM. We can see that the initialization of $\alpha(t)$ in Figure 5b is rather poor. Nevertheless, Zip-dLBM finds the right trend of the mixture proportions over time, effectively emptying the two superfluous clusters. Furthermore, to evaluate the quality of the clustering, we use a measure called CARI, recently introduced by Robert et al. (2021). This new criterion is based on the Adjusted Rand Index (Rand, 1971) and it was developed especially for being applied to co-clustering methods. The closer the index is to 1, the more both the row and column partitions are close to the actual ones, whereas the closer the value is to 0, the greater the difference between the true and estimated labels. In this experiment a CARI index value was calculated for each time instant; the obtained CARI index is 0.98.

Now, as for the second experiment of testing the robustness of the model to initialization, we simulate the data in such a way that in the clusters in line there is one that is empty at the beginning of the period under consideration, then fills up towards the middle of the period, and then empties again at the end, Figure 6a shows this dynamic. Through this experiment we want to show how Zip-dLBM is able to find the right evolution of mixture proportions despite the complex dynamics. Figure 6 depicts the simulated parameters on the left, the initial estimates in the middle, and the final estimates on the right. Looking at the middle part, in Figure 6b, we can see that the initialization process is not particularly helpful to the model because of the switched labels and a poor

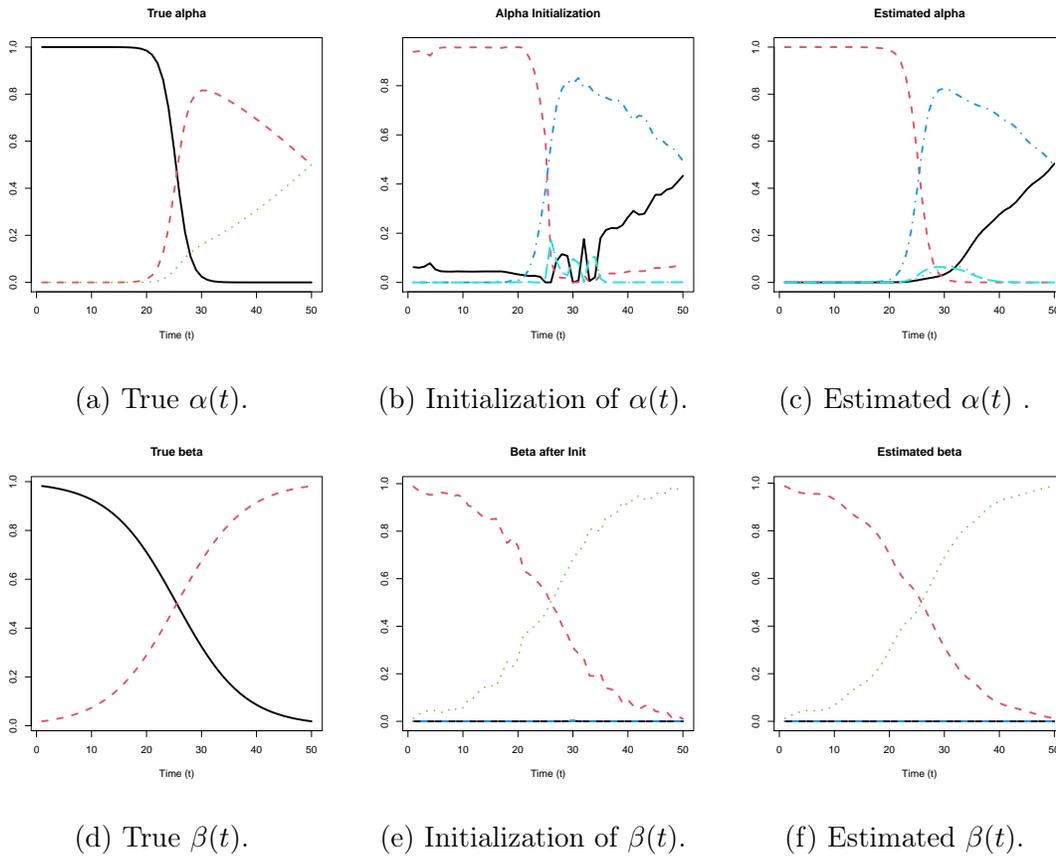


Figure 5: Evolution of the true (left), initialized (center) and estimated (right) proportions of the parameters $\alpha(t)$ and $\beta(t)$, respectively.

parameters estimation. Despite the initialization, we see in Figure 6c that Zip-dLBM is perfectly able to recognize the initially empty cluster which then gradually fills up and empties again. In this experiment, the CARI index, obtained by averaging the indexes over time, is 0.95.

4.3 Model selection experiment

Previous experiments have allowed us to attest that the initialization strategy is globally robust and that the application of Zip-dLBM allows us to correct for poor initializations with respect to the number of clusters in rows or columns. Therefore, in this experiment we test the global capability in choosing the optimal number of clusters in rows and columns over a larger number of simulated datasets through the combination of the initialization procedure and the application of the Zip-dLBM algorithm. Let us recall that, as mentioned in Section 3, the ICL criterion identifies the optimal number of clusters only at one time instant in order to initialize the parameters optimally. Subsequently, Zip-dLBM is run with a higher number of clusters than those identified by ICL. Hence, to validate the performances on the component activation, 50 independent data sets are generated with the setup explained in Section 4.1, with $Q = 3$ row clusters and $L = 2$ column clusters, a level of sparsity varying between 80% and 90% and the other model parameters equal to:

Cluster	α	β
1	0.2 to 0.8	0.1 to 0.99
2	0.18 to 0.14	0.99 to 0.1
3	0.6 to 0.06	-

$$\Lambda = \begin{bmatrix} 6 & 4 \\ 1 & 2 \\ 7 & 3 \end{bmatrix}$$

Then, Zip-dLBM is applied on those simulated data sets using values of Q and L equal to 10. Table 1 shows the percentage of selections. The highlighted cell corresponds to the actual value of Q and L . Zip-dLBM succeeds 86% of the time to identify the correct model. Specifically, to evaluate the results of this experiment, we averaged the

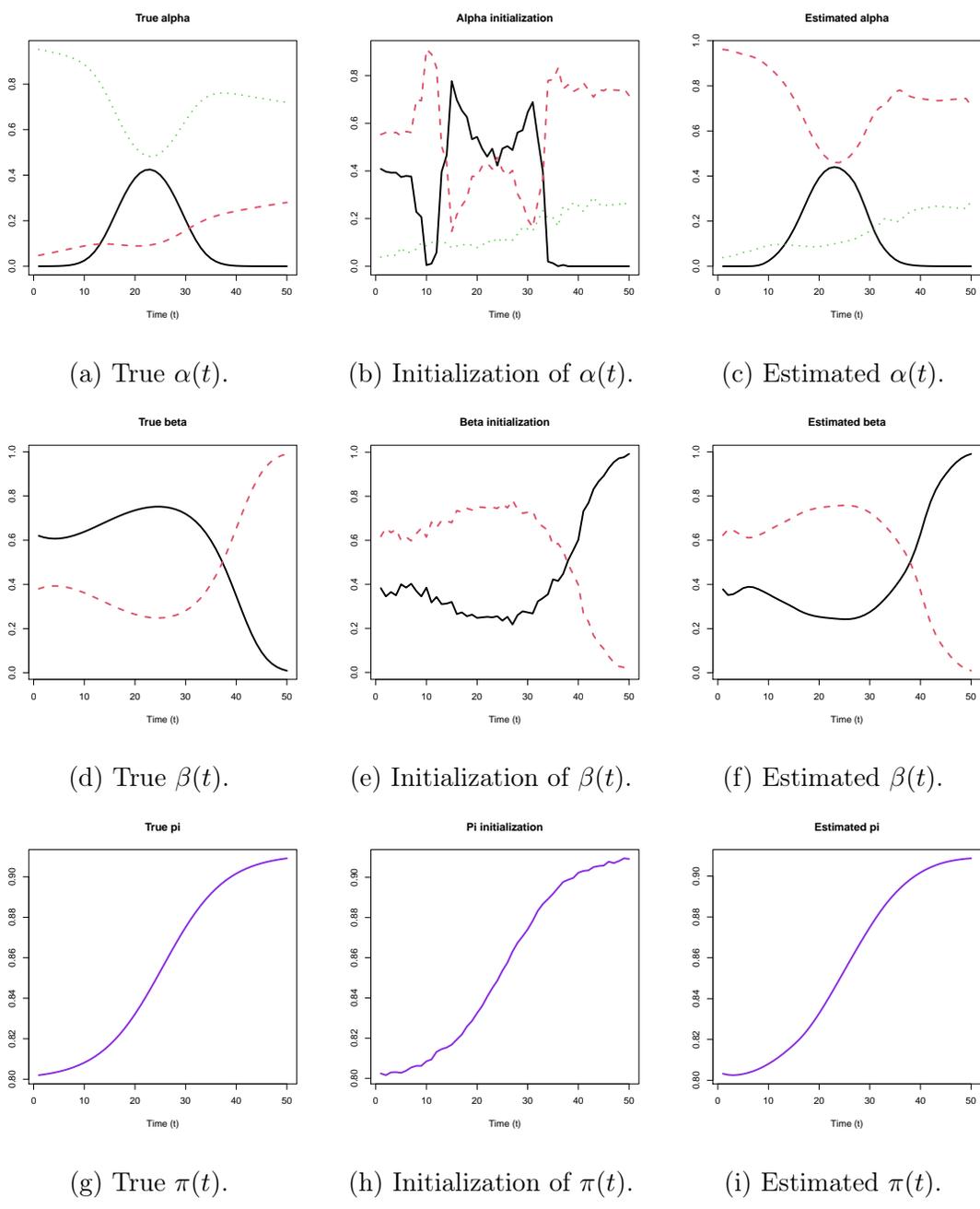


Figure 6: Evolution of the true (left), initialized (center) and estimated (right) proportions of the parameters $\alpha(t)$, $\beta(t)$ and $\pi(t)$, respectively.

membership probability of the two estimated mixing parameters, $\alpha(t)$ and $\beta(t)$; exceeding clusters having an average membership probability of less than $1e-3$ were considered to be off. Among the results of the 50 simulated data sets, we report in Figure 7, as an illustrative example, one of the component activation results. We see that not only the unnecessary clusters remained empty, but also the estimates of the $\alpha(t)$ and $\beta(t)$ are good, as Zip-dLBM manages to identify the evolution of the two mixing parameters over time, despite the number of clusters given as input is not the optimal one.

Q/L	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	86	0	0	0	0	0	0	0	0
4	0	2	0	0	0	0	0	0	0	0
5	0	2	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	4	0	0	0	0	0	0	0	0
9	0	2	0	0	0	0	0	0	0	0
10	0	4	0	0	0	0	0	0	0	0

Table 1: Model selection. Percentage of activated components on 50 simulated data sets. The highlighted cell corresponds to the actual value of Q and L.

4.4 Benchmark study

The goal of this last experiment is to compare Zip-dLBM with two state-of-the-art methods to recover the data structure. First, Zip-dLBM is compared with a model based on the same assumptions but which does not take into account the sparsity modeling over time.

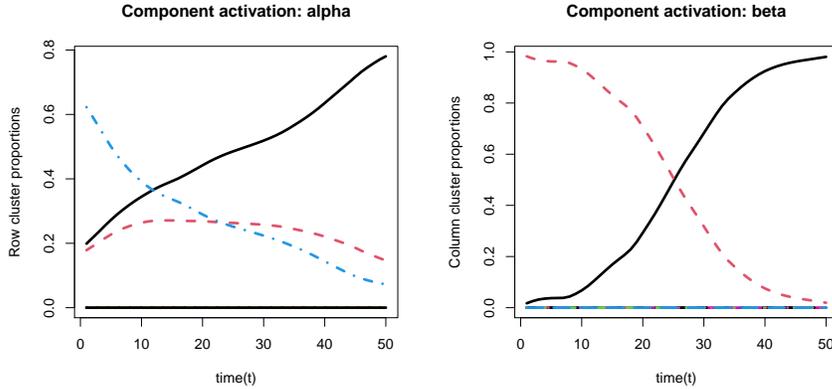


Figure 7: View of a model selection result: the useful clusters are activated while the useless ones lie empty on the basis of the figure.

Denoted by $\text{Zip-dLBM}_{\pi(\cdot)=0}$, the model does not take into account the excess of zeros in the data and it is obtained by setting the sparsity parameter $\pi(t)$, with t in $[0, T]$, equal to zero. The other model Zip-dLBM is compared with is dLBM proposed by Marchello et al. (2022) where not only the sparsity is not taken into account but the cluster memberships Z and W are not time-dependent, i.e. cluster switches are not allowed. However, the expected number of interactions between co-clusters (the parameter Λ) changes in time in dLBM.

We chose to evaluate the results with the CARI index. In order to compare the affectations to the clusters over time, the cluster labels in dLBM were repeated as many times as the number of time instants, and then compared to the affectations of the simulated data using the CARI index. To make this comparison more complete, we defined two simulation scenarios. In Scenario A, the data are simulated as described in Section 4.1 but with a constant sparsity level of 80%, fixed in time. In Scenario B the sparsity evolves in time from 80% to 90%. Table 2 displays the results of this comparison, in terms of average CARI values, reported with standard deviations.

In Scenario A, Zip-dLBM performs well reaching a CARI value of 0.93, on the other

hand Zip-dLBM $_{\pi(\cdot)=0}$ suffers from the excessive number of zeros, whose treatment is not considered, probably affecting the clustering performance. Even worse for dLBM whose CARI index is 0. This is certainly due to the fact that the two latent clustering variables, Z and W , do not evolve over time.

In scenario B, Zip-dLBM performs comparably with the previous scenario, with an average CARI index of 0.94 and a smaller standard deviation. Thus, we see that an increasing level of sparsity does not degrade the performance of the model since it is able to distinguish structural zeros from those coming from the Poisson process. This could even help in improving clustering performance. On the contrary, Zip-dLBM $_{\pi(\cdot)=0}$ performs worse than the results obtained in the scenario A probably due to the increased sparsity in the data.

	Zip-dLBM	Zip-dLBM $_{\pi(\cdot)=0}$	dLBM
Scenario A	0.93 ± 0.13	0.27 ± 0.1	0 ± 0
Scenario B	0.94 ± 0.03	0.16 ± 0.11	0 ± 0.01

Table 2: Co-clustering results for Zip-dLBM, Zip-dLBM $_{\pi(\cdot)=0}$ and dLBM on 50 simulated data according to the two scenarios. Average CARI values are reported with standard deviations.

5 London bike sharing

This section focuses on the application of Zip-dLBM to a large-scale bike-sharing data set in London, with the aim of illustrating the potential of our tool in analyzing and summarizing a real-world sparse and massive data set.

5.1 Protocol and data

The data are collected and publicly distributed by Transport for London¹. We focus on one-month, specifically June 2022, as it represents in our view a neutral choice for the use of shared bikes, both regarding the end of Covid pandemic and the weather conditions. The objective of this application is to analyze how inbound (Arrival) and outbound (Departure) bike rental stations take on different roles, and, consequently, different cluster memberships, depending on the hour of the day. To analyze the data, we summed up the hourly interactions on the working days of the month, in order to obtain a "cumulative" day that we consider from 6 am to 10 pm. So, the time unity measure is one hour and the overall data set is made of by 776,270 observations, for which we consider the departure station name, the arrival station name and the start time date. Moreover, we only consider stations (arrival and departure) that were rented more than 50 times over the month of June 2022. The resulting data set contains 791 departure stations, 791 arrival stations and 16 hours corresponding to 475,586 non-zero entries in the incidence matrix. Figure 8 represents the number of bikes in use in the London sharing system at every hour over the whole month of June. It can be clearly noticed that there are two peaks at the rush hours when people go to or from work (7-8 am and 5-6 pm).

5.2 Summary of the results

We fit Zip-dLBM to the data, on a MacBook Pro, 2020, with a processor of 2,3 GHz Quad-Core Intel Core i7 and 16 GB of RAM. For the initialization, as explained in Section 3.4, we computed the ICL criterion on one data slice, corresponding on the hour 9 am - 10 am, where the optimal number of clusters identified by the model selection criterion are $\hat{Q} = 6$ and $\hat{L} = 6$. Then, we initiated the model parameters through the cascade process described in Algorithm 2 and we ran Zip-dLBM with $Q = 10$ and $L = 10$ to allow the model to fill or empty clusters as needed.

¹<https://cycling.data.tfl.gov.uk>

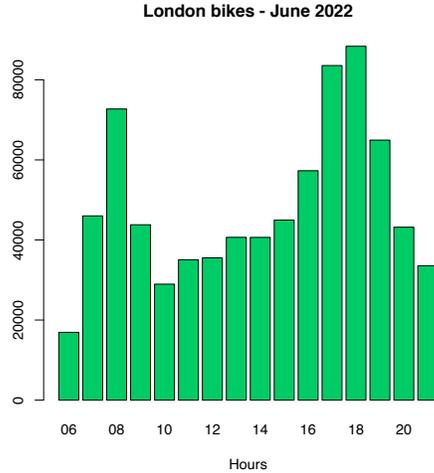


Figure 8: Barplot of the number of bikes taken from the London sharing system, from 6 am to 10 pm, in a cumulative day, corresponding to June 2022.

Figure 9 represents the estimated Poisson intensities Λ for Zip-dLBM. This figure only focuses on the 6 groups of departure stations, denoted by the letter D, and the 6 groups of arrival stations, denoted by the letter A, that have been activated in the inference. Each color refers to a departure (rows) or arrival (columns) cluster and the higher is the value in each block, the strongest is the relationship (i.e the expected number of bikes exchanged in the time unit) between the related pair of clusters. Looking at this figure, it can be seen that some clusters are strongly related whereas others are not at all. For example, cluster D1 (light blue) of the departure stations is highly related with clusters A4 and A6 of arrival stations. Also, cluster D5 of departure stations shows the same behavior but with lower intensity levels. Hence, one may think that the arrival clusters A4 and A6 are located in the city center or in highly busy areas. This intuition is supported by Figure 1 in Appendix A, where an animation showing the evolution of cluster composition and position over the day is depicted. In fact, looking at these figures one can see that cluster D1 of the departure stations is mostly concentrated at central

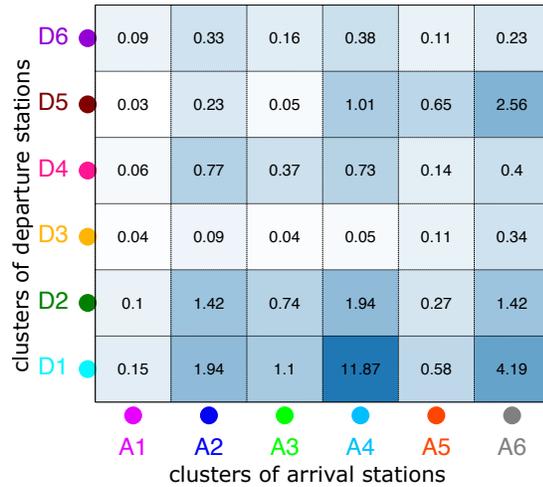


Figure 9: Estimated Poisson intensity function, each color represents a different departure (arrival) cluster.

locations and, more specifically, at the stations emitting the highest number of bikes, such as Hyde Park, King’s Cross, and Queen Elizabeth Olympic Park. We also noted that during the month of June 2022 in Hyde Park and Queen Elizabeth Olympic Park several major events were held, such as the Rolling Stones concert (Hyde Park) and the Red Hot Chili Peppers concert (Queen Elizabeth Olympic Park). This suggests that this cluster includes stations that are subject to a higher-than-normal load. Also, from Figure 1 in Appendix A, we can see a particular behavior of cluster D2 (green) of outgoing bikes. This cluster is characterized by bikes leaving the suburbs during the morning peak hours (7 am-9 am) and then concentrating more in the city center during the day and especially in the evening rush hours (4 pm-6 pm). In addition, this cluster has a strong relationship with clusters A2, A4 and A6 of arrival stations. Among them, clusters A4 and A6 represent mostly central stations, while cluster A2 is concentrated in the city center in the morning rush hours and in the suburbs in the evening rush hours. We

might infer that this dynamic is typical of workers who decide to bike to their workplaces. On the contrary, clusters D3 and D6 of departure stations have a very low intensity of interactions with the arrival clusters, however they are really spread all over the city. In particular, cluster D3 has the highest level of interactions with cluster A6 of arrival stations, while cluster D6 has the highest intensity level with cluster A2 and cluster A4 of arrival stations. The main difference between clusters D3 and D6 concerns their location. In fact, all over the day, there are really few points belonging to cluster D3 and they are mainly concentrated in the Greenwich peninsula and, only at the end of the day, in the city center. Cluster D6, on the contrary, concentrates in the city center and in some specific areas, such as Greenwich peninsula, Wandsworth and Sheperd’s Bush early in the morning, then it spreads all over the city. Therefore, from the large initial data matrix, Zip-dLBM was able to identify consistent and relevant clusters of the London sharing bike. stations

5.3 Interpretation of the estimated parameters

To better understand the results, we now focus on the estimates of the other model parameters. Figure 10 shows the estimated evolution of the sparsity parameter over time. We see that, at the beginning of the day, 6 am, the sparsity is at 95%, then as we approach the morning peak, the number of borrowed sharing bikes increases and consequently the sparsity decreases, reaching 86% at 8 am. Between 9 am and 2 pm, it again increases slightly (90%) and then decreases as we approach the peak at the end of the day, when workers leave work. In fact, at 6 pm the sparsity level reaches its daily minimum at a level of 80%.

Figures 11a and 11b show the estimation of the mixing parameters $\alpha(t)$ and $\beta(t)$. From Figure 11a, we see how cluster D2 has a precise evolution, corresponding to the daily work rhythm. Cluster D4 and Cluster D6 on the contrary, have exactly the opposite behavior, filling up in the non-rush hours of the day, starting at 10 am, then emptying out in the

rush hour of the afternoon and filling up again in the evening. We might infer that the outbound bikes in these clusters are mostly rented by tourists. Cluster D1, on the other hand, as mentioned earlier, is very peculiar because it contains a few stations that emit a lot of bikes, probably when there are special events that mobilize a large number of people. (e.g concerts, football matches, etc.)

Looking at Figure 11b instead, we see estimates of the mixing proportions of the arrival bike clusters. Cluster A2 again has the typical workday evolution, thus including those bike stations taken to get to the workplace and back home at the end of the day. Another particular group is cluster A3. Its proportions increase from 10 am reaching a peak between 2 pm and 3 pm, and while all other clusters tend to empty out in the evening, this one increases. Cluster A5 fills in mid-morning and then remains stable during the rest of the day. Clusters A4 and A6, on the other hand, are very small in terms of proportions. As we saw earlier, they in fact include few but very central stations with a very high density of interactions.

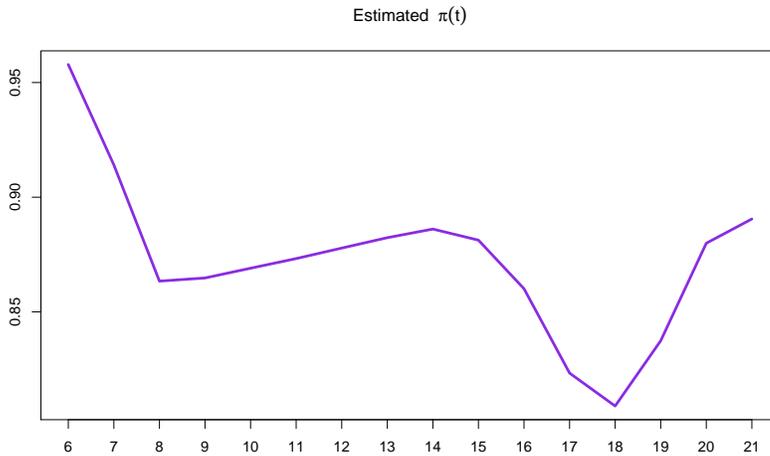
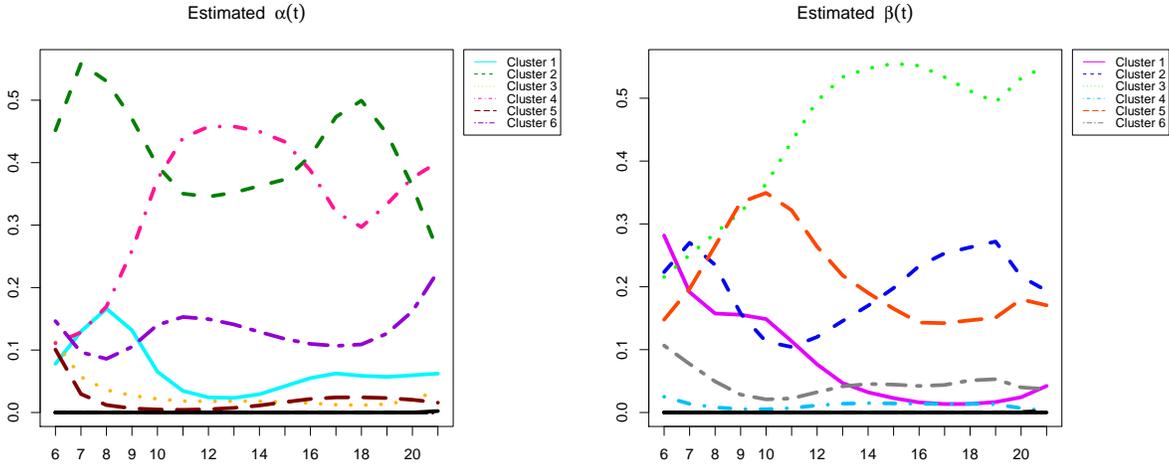


Figure 10: Evolution of $\hat{\pi}(t)$ estimation.



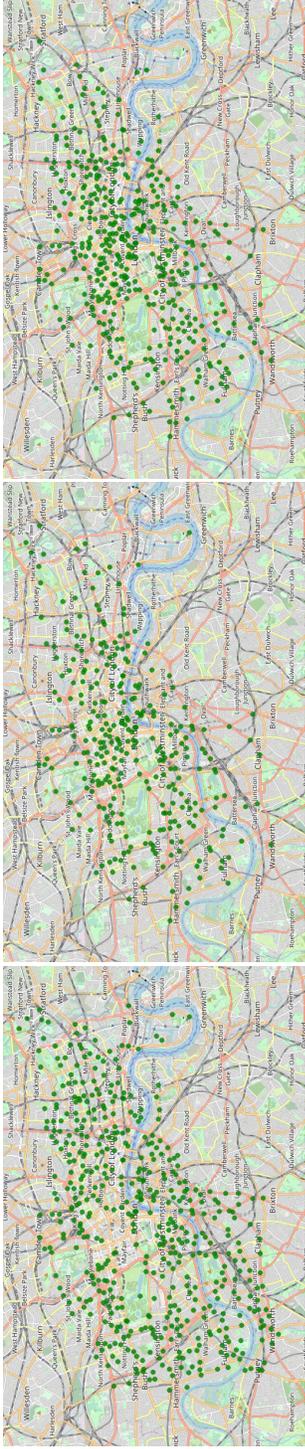
(a) Evolution of the estimates $\hat{\alpha}(t)$.

(b) Evolution of the estimates $\hat{\beta}(t)$.

Figure 11: Evolution of the estimates $\hat{\alpha}(t)$ and $\hat{\beta}(t)$. Each color represents a different cluster.

5.4 Insights into the evolution of two departure and arrival clusters

For pedagogical purposes, in this section we choose two clusters, specifically the departure bike cluster D2 and the arrival bike cluster A2, by analyzing the results in detail. Figure 12 depicts cluster D2 of the departing stations and its evolution in 3 time instants: 7 am, 1 pm and 6 pm. As mentioned, given its specific dynamics, this cluster is likely to be populated by bicycle stations used mostly by workers. In fact, in Figure 12a we see that at 7 am there are many points, mostly distributed outside the city center. Thereafter, around 1 pm (Figure 12b) the volume of points decreases. Then, toward the afternoon rush hour, in Figure 12c, the density of points increases again. Indeed, in Figure 12c we see that most of the points are now located in the center. This behavior is typically due to the workers from the suburbs going to work in the city center while at the end of the day taking the shared bikes back to leave the city center.



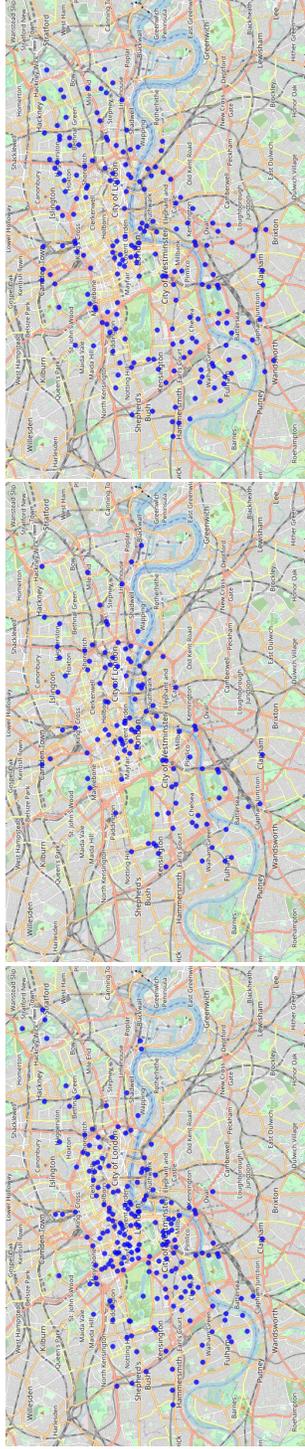
(a) Cluster D2 - 7 am.

(b) Cluster D2 - 1 pm.

(c) Cluster D2 - 6 pm.

Figure 12: Snapshots of the evolution of the departure cluster D2 at three different times in the day: 7 am, 1 pm, 6 pm.

43



(a) Cluster A2 - 7 am.

(b) Cluster A2 - 1 pm.

(c) Cluster A2 - 6 pm.

Figure 13: Snapshots of the evolution of the arrival cluster A2 at three different times in the day: 7 am, 1 pm, 6 pm.

Similarly, Figure 13 shows the cluster A2 of the arrival stations and its evolution in 3 time instants: 7 am, 1 pm and 6 pm. Here we note how at 7 am cluster A2 contains arrival stations all located in the city center. In contrast, in Figure 13b, the number of bikes belonging to this cluster decreases and the locations are more scattered. Whereas, in Figure 13c, we note how at the end of the workday, most of the arrival stations in this cluster are no longer located in the center but in the suburbs. Thus, from the comparison of Figure 12 and Figure 13 we notice a complementary trend, dictated by the fact that the arrival and departure stations in the two selected clusters are both characteristic of the daily work schedules.

6 Conclusion

This work is born out of the need to analyze and summarize observations and features of a dynamic matrix in a simultaneous way. We have proposed a dynamic co-clustering technique, with the purpose of simultaneously performing clustering of rows and columns along the time dimensions. Since observations and features are allowed to change their cluster memberships in time, it is of great interest to look for structural changes in the way clusters interact with each other along the considered time period. We have introduced a generative zero-inflated dynamic latent block model, that can be further adapted to several zero-inflated probability distributions. For ease of reading the mathematical and experimental part, in this paper we used the Zero-Inflated Poisson distribution, thus introducing the Zero-Inflated Poisson Dynamic Latent Block model (Zip-dLBM). The time modeling relies on three systems of ordinary differential equations. Inference is done using a Variational EM algorithm together with stochastic optimization for the parameters of the dynamic systems. The performance of our approach, called Zip-dLBM in the Poisson case, is evaluated through applications to several simulated data scenarios and compared with competing methods. Then, Zip-dLBM was fitted to a large-scale real data set, the London sharing bikes. In this context, Zip-dLBM provided meaningful and interpretable

results. As a further work, it would be of interest to develop an online inference algorithm for Zip-dLBM, to be used as a real-time co-clustering tool. It would also be interesting to do some real-time change point detection, with possible applications to data from several domains (transportation or medical, for instance). A further idea would be to introduce dynamic systems also for the variational parameters, in the inference model. However, this choice might be very computationally expensive and require further investigation.

Acknowledgements

This work has been supported by the French government, through the 3IA Côte d'Azur, Investment in the Future, project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

References

- Ailem, M., Role, F., and Nadif, M. (2017). Sparse poisson latent block model for document clustering. *IEEE Transactions on Knowledge and Data Engineering*, 29(7):1563–1576.
- Bergé, L. R., Bouveyron, C., Corneli, M., and Latouche, P. (2019). The latent topic block model for the co-clustering of textual interaction data. *Computational Statistics & Data Analysis*, 137:247–270.
- Biernacki, C., Celeux, G., and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE transactions on pattern analysis and machine intelligence*, 22(7):719–725.
- Bishop, C. M. (2006). Approximate inference. pages 461–517. Springer-Verlag, Berlin, Heidelberg.
- Boutalbi, R., Labiod, L., and Nadif, M. (2020). Tensor latent block model for co-clustering. *International Journal of Data Science and Analytics*, pages 1–15.
- Boutalbi, R., Labiod, L., and Nadif, M. (2021). Implicit consensus clustering from multiple graphs. *Data Mining and Knowledge Discovery*, 35(6):2313–2340.
- Bouveyron, C., Bozzi, L., Jacques, J., and Jollois, F.-X. (2018). The functional latent block model for the co-clustering of electricity consumption curves. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 67(4):897–915.
- Bouveyron, C., Celeux, G., Murphy, T. B., and Raftery, A. E. (2019). *Model-based clustering and classification for data science: with applications in R*, volume 50. Cambridge University Press.
- Casa, A., Bouveyron, C., Erosheva, E., and Menardi, G. (2021). Co-clustering of time-dependent data via the shape invariant model. *Journal of Classification*, 38(3):626–649.

- Corneli, M., Bouveyron, C., and Latouche, P. (2020). Co-clustering of ordinal data via latent continuous random variables and not missing at random entries. *Journal of Computational and Graphical Statistics*, pages 1–15.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.
- Dhillon, I. S. (2001). Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–274.
- Dhillon, I. S., Mallela, S., and Modha, D. S. (2003). Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98.
- Ding, C., Li, T., Peng, W., and Park, H. (2006). Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135.
- Gent, C. and Sheppard, C. (1992). Special feature. predicting time series by a fully connected neural network trained by back propagation. *Computing & Control Engineering Journal*, 3(3):109–112.
- Govaert, G. and Nadif, M. (2003). Clustering with block mixture models. *Pattern Recognition*, 36(2):463–473.
- Govaert, G. and Nadif, M. (2008). Block clustering with bernoulli mixture models: Comparison of different approaches. *Computational Statistics & Data Analysis*, 52(6):3233–3245.

- Govaert, G. and Nadif, M. (2010). Latent block model for contingency table. *Communications in Statistics - Theory and Methods*, 39(3):416–425.
- Jaakkola, T. S. and Jordan, M. I. (1997). A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, pages 283–294. PMLR.
- Jacques, J. and Biernacki, C. (2018). Model-based co-clustering for ordinal data. *Computational Statistics & Data Analysis*, 123:101–115.
- Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1998). An introduction to variational methods for graphical models. In *Learning in graphical models*, pages 105–161. Springer.
- Keribin, C., Brault, V., Celeux, G., and Govaert, G. (2015). Estimation and selection for the latent block model on categorical data. *Statistics and Computing*, 25(6):1201–1216.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Labioud, L. and Nadif, M. (2011). Co-clustering under nonnegative matrix tri-factorization. In *International Conference on Neural Information Processing*, pages 709–717. Springer.
- Lambert, D. (1992). Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, 34(1):1–14.
- Li, N., Elashoff, D. A., Robbins, W. A., and Xun, L. (2011). A hierarchical zero-inflated log-normal model for skewed responses. *Statistical Methods in Medical Research*, 20(3):175–189.
- Lindstrom, M. J. (1995). Self-modelling with random shift and scale parameters and a free-knot spline shape function. *Statistics in medicine*, 14(18):2009–2021.

- Lomet, A. (2012). *Sélection de modèle pour la classification croisée de données continues*. PhD thesis, Compiègne.
- Marchello, G., Fresse, A., Corneli, M., and Bouveyron, C. (2022). Co-clustering of evolving count matrices with the dynamic latent block model: application to pharmacovigilance. *Statistics and Computing*, 32(3):1–22.
- Matias, C. and Miele, V. (2017). Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(4):1119–1141.
- Nowicki, K. and Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association*, 96(455):1077–1087.
- Ospina, R. and Ferrari, S. L. (2012). A general class of zero-or-one inflated beta regression models. *Computational Statistics & Data Analysis*, 56(6):1609–1623.
- Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.
- Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.
- Ridout, M., Hinde, J., and Demétrio, C. G. (2001). A score test for testing a zero-inflated poisson regression model against zero-inflated negative binomial alternatives. *Biometrics*, 57(1):219–223.
- Robert, V., Vasseur, Y., and Brault, V. (2021). Comparing high-dimensional partitions with the Co-clustering Adjusted Rand Index. *Journal of Classification*, 38:158–186.
- Selosse, M., Jacques, J., and Biernacki, C. (2020). Model-based co-clustering for mixed type data. *Computational Statistics & Data Analysis*, 144:106866.

Yang, T., Chi, Y., Zhu, S., Gong, Y., and Jin, R. (2011). Detecting communities and their evolutions in dynamic social networks—a bayesian approach. *Machine learning*, 82(2):157–189.