Deep dynamic co-clustering of count data streams: application to pharmacovigilance

Giulia Marchello, Alexandre Destere, Marco Corneli, Charles Bouveyron

Deep dynamic co-clustering of count data streams: application to pharmacovigilance

Giulia Marchello[1], Alexandre Destere[2], Marco Corneli[1,3] & Charles Bouveyron[1]

[1] *Université Côte d'Azur, Inria, CNRS, Laboratoire J.A.Dieudonné, Maasai team, Nice, France.*
[2] *Université Côte d'Azur Medical Centre, Department of Clinical Pharmacology, Nice, France* [3] *Université Côte d'Azur, Laboratoire CEPAM, Nice, France.*

Abstract. Co-clustering is a widely used technique that allows the analysis of complex and high-dimensional data in various domains. However, existing models mostly concentrate on continuous and dense data in fixed time situations, where cluster assignments remain unchanged over time. For example, in the field of pharmacovigilance, it is crucial to cluster in real time drugs and adverse effects simultaneously, facilitating the automation of safety signal detection processes. However, traditional co-clustering methods require all the data to be loaded into memory, which can be a challenge for large datasets or even impossible in certain scenarios. The proposed online co-clustering model is designed to overcome this challenge by processing the data incrementally, one step at a time. This work introduces a novel inference process for the latent block model that addresses the challenge of online co-clustering of sparse data matrices. 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 we make use of a Bayesian online change point detection method on both cluster memberships and data sparsity estimations. The inference relies on an original variational procedure whose maximization step trains a LSTM neural network in order to solve the dynamical systems. Numerical experiments on simulated datasets demonstrate the effectiveness of the proposed methodology in the context of count data streams. Then, we fit the model to a large-scale dataset supplied by the Regional Center of Pharmacovigilance of Nice (France), providing meaningful online segmentation of drugs and adverse drug reactions.

Keywords. Co-clustering, change point detection, zero-inflated distributions, online inference, VEM algorithm, data streams

1 Introduction

The use of unsupervised machine learning techniques, such as clustering, to summarize data is increasingly needed. A topical application area is pharmacovigilance, whose main activity concerns in gathering drug associated adverse events to detect safety signals about drugs. Currently, the detection of safety signals heavily relies on manual expert analysis, leading to potential incompleteness due to the workload involved and the need for substantial data before critical events can be detected. Such methods would enable pharmacovigilance experts to focus on searching for ADRs in all the digital documents and reports generated by healthcare establishments. Consequently, there is a pressing need to develop automated methods for safety signal detection in pharmacovigilance. Clustering techniques can help in this task to effectively summarize data to detect safety signals along the time. In addition to pharmacovigilance, clustering methods are increasingly important in various domains such as social media, e-commerce, biomedical data, finance, and genetics. Traditional clustering approaches may struggle with high-dimensional sparse datasets, making co-clustering an appealing alternative. Co-clustering is useful in this context since it clusters both observations and features simultaneously, providing useful data summaries. Although many notable methods have been introduced in this field in recent decades, the development of dynamic co-clustering methods remains largely unexplored. Also, in many applications and particularly in the case of pharmacovigilance, the data generating process can change over time, which can result in changes in the clustering patterns. For example, drugs may have unexpected adversarial effects in response to changes in the molecular principle, or new drugs may be introduced to the market. Online change point detection algorithms can detect when the data generating process has changed and trigger further investigations. In this paper, we address this challenge by exploring the development of an online model-based co-clustering tool for real-time safety signal detection. By treating adverse drug reaction (ADR) notifications as count data observed over time, our approach allows for the identification of temporal breaks in the safety signals. This facilitates the creation of alerts and provides room for further investigation by medical authorities. The primary objective of this research is to showcase the potential of our proposed method as a routine tool in pharmacovigilance.

1.1 Related Work

This section summarizes the related work in dynamic co-clustering and change point detection.

Co-clustering and Latent Block Models Co-clustering is a valuable tool for analyzing datasets as it allows for the simultaneous clustering of both observations and features. There are various types of co-clustering methods, that 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 robust statistical foundations and adaptability to various data types and levels of sparsity. 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 into 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, in the same paper, is also implemented for different types of datasets: continuous data (Gaussian TLBM), binary data (Bernoulli TLBM) and contingency tables (Poisson TLBM).

Dynamic models for clustering and co-clustering While there has been an extensive body of literature, over the past decade, on static model-based clustering and co-clustering techniques the use of dynamic models in this context is a more recent development. It is worth noting that a more plentiful 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, which does not need that data matrices to be 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. (2022b) 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 extend LBM to the dynamic case, this model has the limitation of not allowing cluster switches of rows/columns. Later the same authors proposed Zip-dLBM (Marchello et al., 2022a), a further dynamic extension of LBM. In Zip-dLBM, the authors allow observations and features to change clusters over time, this paper has been proposed in a general way for any Zero-Inflated distribution, to model highly sparse scenarios, typical of interaction datasets. In a different framework, Casa et al. (2021) extended 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, while being a first step toward the dynamic expansion of co-clustering methods, these methods can be used mostly for macroscopic analysis, as they do not allow for the temporal dependence of variables.

Change point detection Change points refer to sudden shifts in the pattern of a time series dataset. Identifying these points can be valuable in analyzing and forecasting time series data. Also, these algorithms are specifically designed to detect the precise moment when a process that is evolving over time has undergone a significant change. This specific moment indicates a shift in the underlying process generating the observed data points. Several change point detection algorithms have been proposed in the literature (e.g. Basseville et al. (1993); Cheng and Thaga (2005); Van den Burg and Williams (2020); Montgomery (2020)). These algorithms can be categorized into two types: online and offline. Online algorithms are designed to operate in real-time, with upcoming time series data. In contrast, offline algorithms are intended to run after the entire dataset has been collected. Since the goal here is to detect on the fly pharmacovigilance events, we focus hereafter on online methods only. Regarding online change point detection algorithms, recent studies have shown that among the most effective methods there are the likelihood and probabilistic approaches (Kondratev et al., 2022; Kavitha and Punithavalli, 2010). On this subject, a seminal paper was proposed by Adams and MacKay (2007) introducing the Bayesian Online Change Point Detection (BOCD). The idea of BOCD is to identify change points using the so-called run lengths or segment. Whenever a new data point becomes available, the algorithm determines the likelihood of the corresponding run length increasing by one. If the probability of change is higher than that of growth, the run length resets to zero, and a change point is identified. Alternatively, another approach has been proposed by Kawahara and Sugiyama (2009), introducing subspace identification for change point detection. Here, the change points are detected by estimating a state-space model behind time series.

1.2 Our contribution

This paper proposes an online extension of Zip-dLBM (Marchello et al., 2022a), a co-clustering technique designed to handle time-evolving data matrices that may contain many empty entries. We aim to introduce three novelties in this regard. The first is the ability of the estimation algorithm to work online, with streams of data. The second is the addition of an online change point detection method. By capturing the data dynamic behavior, the method can identify abnormal events that affect the generative process. To detect these changes we make use of the Bayesian Online Change Point Detection (BOCD, Adams and MacKay, 2007) that runs on the estimated model parameters in real time. The algorithm iteratively updates the posterior probabilities of the change points, based on the data observed so far. The third novelty relies on a different modeling choice for the time evolving parameter. In fact fully connected neural networks are substituted with LSTMs, as their

structure is deemed more appropriate for the purpose. Therefore, this model introduces a new approach by incorporating an online inference method for Zip-dLBM and online changing point detection.

1.3 Organization of the paper

This paper is organized as follows. Section 2 recalls the generative model Zip-dLBM. Section 3 introduces the proposed online inference for stream data. Section 4 presents various experiments on simulated data to test and evaluate the model performances. In Section 5, an application on a real ADRs dataset is presented to illustrate the potential of Stream Zip-dLBM in pharmacovigilance. Section 6 provides some concluding remarks.

2 Stream Zip-dLBM

The following section reminds the Zero-Inflated Poisson Dynamic Latent Block Model (Zip-dLBM, Marchello et al., 2022a) for batch processing. Notice that, although this model has been proposed in a general way for any Zero-Inflated distribution, we focus here on the Zero-Inflated Poisson (ZIP) distribution, since count data are considered in pharmacovigilance. In Zip-dLBM the observed data are assumed to be collected into time evolving matrices, over the interval $[0, T]$. Being in a discrete time setting, we assume a time partition of equally spaced points:

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

With a slight abuse of notation we denote by t the time point t_u. At time t, the incidence matrix $X(t) \in \mathbb{N}^{N \times M}$ has $X_{ij}(t)$ as generic element that counts the number of interactions between the observation i and feature j that took place between $t-1$ and t. The rows of $X(t)$ are indexed by $i = \{1, ..., N\}$ and the columns by $j = \{1, ..., M\}$. The goal of Zip-dLBM is to cluster both the rows and columns of a series of data matrices, $\{X(t)\}_t$, with $t \in [0, T]$.

2.1 A Zero-Inflated Dynamic Latent Block Model

First, let us now consider that we have a time interval of fixed lenght, $[0, T]$, and that the model does not run online. The goal of Zip-dLBM is to cluster both the rows and columns of a series of data matrices, $\{X(t)\}_t$, that change over time. The numbers of clusters for rows and columns are represented by Q and L, respectively, and remain constant over time. However, the cluster membership of each row and column is allowed to change in $[0, T]$.

2.1.1 Clusters modeling

To track the changes in cluster memberships over time, we use two evolving random matrices, $Z(t)$ and $W(t)$, for rows and columns, respectively. With $Z(t) \in \{0, 1\}^{N \times Q}$, its the i-th row

is denoted by $Z_i(t)$ and $W(t) \in \{0,1\}^{M\times L}$, its j-th row is denoted by $W_j(t)$. The two distributions are respectively parametrized by $\alpha(t)$ and $\beta(t)$. More formally:

$$Z_i(t) \sim \mathcal{M}(1, \alpha(t) := (\alpha_1(t), \ldots, \alpha_Q(t))), \qquad \text{independently } \forall i, \forall t \in \{0, \ldots, T\}, \tag{1}$$

where $\mathcal{M}(\cdot,\cdot)$ denotes the multinomial probability mass function and $\alpha_q(t) = \mathbb{P}\{z_{iq}(t) = 1\}$, with $\sum_{q=1}^{Q} \alpha_q(t) = 1$. Thus $Z(t) := \{z_{iq}(t)\}_{i\in 1,\ldots,N; q\in 1,\ldots,Q}$ 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:

$$W_j(t) \sim \mathcal{M}(1, \beta(t) := (\beta_1(t), \ldots, \beta_L(t))), \qquad \text{independently } \forall j, \forall t \in \{0, \ldots, T\}, \tag{2}$$

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.

2.1.2 Sparsity modeling

In order to model a potentially extreme sparsity, the observed data are assumed to follow a mixture of block-conditional Zero-Inflated Poisson (ZIP) distributions, where the entries $X_{ij}(t)$ are conditionally independent in (i, j, t):

$$X_{ij}(t)|Z_i(t), W_j(t) \sim ZIP(\Lambda_{Z_i(t),W_j(t)}, \pi(t)) \ , \tag{3}$$

where Λ is the $Q \times L$ block-dependent intensity function of the Poisson component, and $\pi(t)$ is a vector of length T that indicates the level of sparsity at any given time period. In order to ease the inference, we finally provide an equivalent formulation of the above equations in terms of a hidden random matrix, $A \in \{0,1\}^{N\times M}$, where, independently for all i and j:

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

with $\mathcal{B}(\cdot)$ denoting the Bernoulli distribution of parameter $\pi(t)$ and such that:

$$\begin{aligned} &A_{ij}(t) = 1 \Rightarrow X_{ij}(t)|Z_i(t), W_j(t) = 0 \\ &A_{ij}(t) = 0 \Rightarrow X_{ij}(t)|Z_i(t), W_j(t) \sim \mathcal{P}(X_{ij}(t), \Lambda_{Z_i(t),W_j(t)}), \end{aligned} \tag{4}$$

where $\mathcal{P}(\cdot, \Lambda)$ denotes the Poisson distribution with intensity parameter Λ.

2.1.3 Modeling the temporal evolution of the parameters

We assume that the evolution of the mixing proportions α, β, and the sparsity parameter π are governed by a system of ordinary differential equations (ODEs). By using ODEs, we can

model the temporal evolution of both the composition of clusters and sparsity. Since we work in discrete time we discretize the dynamic systems by making use of their Euler schemes:

$$\begin{cases} a(t+1) = a(t) + f_Z(a(t)), \\ b(t+1) = b(t) + f_W(b(t)), \\ c(t+1) = c(t) + f_A(c(t)), \end{cases} \tag{5}$$

where f_Z, f_W and f_A are assumed to be three continuously differentiable functions and:

$$\begin{cases} \alpha_q(t) = \text{softmax}(a_q(t)) = e^{a_q(t)} / \sum_{q=1}^{Q} e^{a_q(t)}, \\ \beta_\ell(t) = \text{softmax}(b_\ell(t)) = e^{b_\ell(t)} / \sum_{\ell=1}^{L} e^{b_\ell(t)}, \\ \pi(t) = e^{c(t)} / 1 + e^{c(t)}. \end{cases} \tag{6}$$

2.2 The joint distribution

The set of the model parameters is denoted by $\theta = (\Lambda, \alpha, \beta, \pi)$ and the latent variables used so far are Z, W, and A, where we denote $\alpha \triangleq \{\alpha(t)\}_t$, $\beta \triangleq \{\beta(t)\}_t$, $\pi \triangleq \{\pi(t)\}_t$. Thus, the likelihood of the complete data reads:

$$p(X, Z, W, A|\theta) = p(X|Z, W, A, \Lambda, \pi)p(A \mid \pi)p(Z|\alpha)p(W|\beta). \tag{7}$$

The terms on the right hand side of the above equation can be further developed, as in Marchello et al. (2022a).

3 Online inference for stream data

In this section, we present the online extension of the Zip-dLBM method, called Stream Zip-dLBM. The objective is to perform co-clustering of rows and columns in real-time as new data become available. To prevent memory overload, we have revisited the original inference algorithm of Zip-dLBM, enabling the data to be processed without the need to store it in memory. To allow the algorithm to update the parameter estimates continuously as a new data is incorporated, we use a moving window, $G_d(t)$, of size d. In more detail, at time t, we keep in memory only the data in the interval $[t-d, t]$, namely $X(t-d), X(t-d+1), \ldots, X(t)$, that will be used for the estimation of the model parameters. The data outside the interval can be discarded to prevent memory overloads and maintain the algorithm's functionality. Once a time point t quit the time window (after passing through it) the parameter estimates at that point become fixed, and the "past" data can be discarded by the inference procedure.

3.1 Variational Assumption

The traditional approach to estimate the model parameters involves maximizing the log-likelihood $p(X|\theta)$. However, in our case a direct maximization is not feasible. Neither we can adopt the EM-algorithm, because the joint conditional distribution of the latent variables

$p(Z, W, |X, \theta)$ cannot be computed, due to the interdependent double missing structure of (Z, W) and their continuous evolution. Additionally, we cannot update α, β and π using closed formulas because of their connection with systems of ordinary differential equations. Therefore, we use a combination of Variational-EM inference and Stochastic Gradient Descent (SGD) to estimate the parameters. Let us thus introduce a variational distribution $q(.)$, over (A, Z, W), in order to decompose the observed data log-likelihood as follows:

$$\log p(X|\theta) = \mathcal{L}(q, \theta) + KL(q(.)||p(.|X, \theta)),$$

where $\mathcal{L}$ denotes a lower bound of $p(X|\theta)$ and is defined as:

$$\begin{aligned}\mathcal{L}(q, \theta) &= \sum_Z \sum_W \sum_A q(Z, W, A) \log \frac{p(X, A, Z, W|\theta)}{q(Z, W, A)} \\ &= E_{q(A,Z,W)}\Big[\log \frac{p(X, A, Z, W|\theta)}{q(A, Z, W}\Big] \\ &= E_{q(A,Z,W)}[\log(p(X, A, Z, W|\theta)] - E_{q(A,Z,W)}[\log(q(A, Z, W))],\end{aligned} \tag{8}$$

and KL indicates the Kullaback-Liebler divergence between the true and the approximate posterior $q(\cdot)$:

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

Now, the objective is to find a distribution $q(.)$ 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 for all t:

$$q(A(t), Z(t), W(t)) = q(A(t))q(Z(t))q(W(t)) = \prod_{i=1}^{N}\prod_{j=1}^{M} q(A_{ij}(t)) \prod_{i=1}^{N} q(Z_i(t)) \prod_{j=1}^{M} q(W_j(t)). \tag{9}$$

3.2 Variational E-Step

The optimal variational updates of $q(\cdot)$, under the assumption in Eq. (9), can be obtained as in Bishop (2006, Ch.10). Denoting by $\delta_{ij}(t) := q(A_{ij}(t) = 1)$ the variational probability of success for $A_{ij}(t)$, the optimal update is:

$$\delta_{ij}(t) = \frac{\exp(R_{ij}(t))}{1 + \exp(R_{ij}(t))}, \tag{10}$$

with:

$$\begin{aligned}R_{ij}(t) = \log(\pi(t)\mathbf{1}_{\{X_{ij}(t)=0\}}) + \sum_{q=1}^{Q}\sum_{\ell=1}^{L}\Big[&- E_{q(W,Z)}[Z_{iq}(t)]E_{q(W,Z)}[W_{j\ell}(t)]X_{ij}(t)\log\Lambda_{q\ell} + \\ &+ E_{q(W,Z)}[Z_{iq}(t)]E_{q(W,Z)}[W_{j\ell}(t)]\Lambda_{q\ell}\Big] + \log X_{ij}(t)! - \log(1 - \pi(t)).\end{aligned}$$

Note that, formally, when $X_{ij}(t) \neq 0$, $R_{ij}(t) = -\infty$ and $\delta_{ij}(t) = 0$, which makes sense: non-null observations in X come from a Poisson distribution with probability one (see Eq. (4)).

Regarding $q(Z)$, let us denote by $\tau_{iq}(t) := q(Z_{iq}(t) = 1)$ the variational probability of success of $Z_{iq}(t)$. The optimal update can be written as:

$$\tau_{iq}(t) = \frac{r_{iq}(t)}{\sum_{q_0=1}^{Q} r_{iq_0}(t)}, \tag{11}$$

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

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

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:

$$\eta_{j\ell}(t) = \frac{s_{j\ell}(t)}{\sum_{\ell_o=1}^{L} s_{j\ell_o}(t)}, \tag{12}$$

where :

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

The proofs of Equations 10, (11) and (12) are provided in Marchello et al. (2022a). It is worth noting that these update equations can be executed step by step, allowing for incremental updates of the variational parameters. Also, note that the update in these equations can be computed independently for any pair (i, j), at any time point t.

3.3 Online variational M-Step

While the updates in the E-step for $\tau(t)$, $\eta(t)$, and $\delta(t)$ depend solely on the current time instant t, the same cannot be said for the updates in the M-step. The M-step involves updating the model parameters, such as $\theta = (\Lambda,\ \alpha,\ \beta,\ \ \pi)$, based on the current estimates obtained in the E-step. In order to obtain the updates 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 $\theta = (\Lambda, \alpha, \beta, \pi)$, while holding the variational distribution $q(\cdot)$ fixed. Denoting t as the current time instant,

we develop Eq.(8) such that the variational lower bound $\mathcal{L}(q,\theta)$ can be written as:

$$\begin{aligned}\mathcal{L}(q,\theta) = &\sum_{u=1}^{t}\sum_{i=1}^{N}\sum_{j=1}^{M}\Bigg\{\delta_{ij}(u)\log(\pi(u)\mathbf{1}_{\{X_{ij}(u)=0\}}) + (1-\delta_{ij}(u))\Bigg[\log(1-\pi(u)) + \\ &+\sum_{q=1}^{Q}\sum_{\ell=1}^{L}\Big\{\tau_{iq}(u)\eta_{j\ell}(u)X_{ij}(u)\log\Lambda_{q\ell} - \tau_{iq}(u)\eta_{j\ell}(u)\Lambda_{q\ell}\Big\}\Bigg] - (1-\delta_{ij}(u))\log(X_{ij}(u)!)\Bigg\} \\ &+\sum_{u=1}^{t}\sum_{i=1}^{N}\sum_{q=1}^{Q}\tau_{iq}(u)\log(\alpha_q(u)) + \sum_{u=1}^{t}\sum_{j=1}^{M}\sum_{\ell=1}^{L}\eta_{j\ell}(u)\log(\beta_\ell(u)) - \sum_{u=1}^{t}\sum_{i=1}^{N}\sum_{q=1}^{Q}\tau_{iq}(u)\log(\tau_{iq}(u)) + \\ &-\sum_{u=1}^{t}\sum_{j=1}^{M}\sum_{\ell=1}^{L}\eta_{j\ell}(u)\log(\eta_{j\ell}(u)) - \sum_{u=1}^{t}\sum_{i=1}^{N}\sum_{j=1}^{M}\Big(\delta_{ij}(u)\log(\delta_{ij}(u)) + (1-\delta_{ij}(u))\log(1-\delta_{ij}(u))\Big).\end{aligned} \tag{13}$$

3.3.1 Update of Λ

Here our goal is to derive the online 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 at every time instant t, to obtain its update, $\hat{\Lambda}$. To find the optimal update we compute the derivative of the lower bound $\mathcal{L}(q,\theta)$ in Eq. (13) with respect to Λ and set it equal to zero, as follows:

$$\begin{aligned}&\frac{\partial\log\mathcal{L}(q,\theta)}{\partial\Lambda_{q\ell}} = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}(1-\delta_{ij}(u))\Big[\frac{\tau_{iq}(u)\eta_{j\ell}(u)X_{ij}(u)}{\Lambda_{q\ell}} - \tau_{iq}(u)\eta_{j\ell}(u)\Big] = 0 \\ \Leftrightarrow &\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}(1-\delta_{ij}(u))\Big[\tau_{iq}(u)\eta_{j\ell}(u)X_{ij}(u) - \tau_{iq}(u)\eta_{j\ell}(u)\Lambda_{q\ell}\Big] = 0 \\ \Leftrightarrow &\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}(1-\delta_{ij}(u))\tau_{iq}(u)\eta_{j\ell}(u)\Lambda_{q\ell} = \sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}\tau_{iq}(u)\eta_{j\ell}(u)\Big[X_{ij}(u) - X_{ij}(u)\delta_{ij}(u)\Big] \\ \Rightarrow &\hat{\Lambda}_{q\ell} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}\tau_{iq}(u)\eta_{j\ell}(u)\Big(X_{ij}(u) - \delta_{ij}(u)X_{ij}(u)\Big)}{\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{t}\tau_{iq}(u)\eta_{j\ell}(u)\Big(1-\delta_{ij}(u)\Big)}.\end{aligned} \tag{14}$$

Although the update of $\hat{\Lambda}_{q\ell}$ sums over all the past observations $(X_{ij}(1),\ldots,X_{ij}(t))$, we can develop Eq. (14) as follows:

$$\begin{aligned}\hat{\Lambda}_{q\ell} &= \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{(t-1)}\tau_{iq}(u)\eta_{j\ell}(u)\Big(X_{ij}(u) - \delta_{ij}(u)X_{ij}(u)\Big) + \sum_{i=1}^{N}\sum_{j=1}^{M}\tau_{iq}(t)\eta_{j\ell}(t)\Big(X_{ij}(t) - \delta_{ij}(t)X_{ij}(t)\Big)}{\sum_{i=1}^{N}\sum_{j=1}^{M}\sum_{u=1}^{(t-1)}\tau_{iq}(u)\eta_{j\ell}(u)\Big(1-\delta_{ij}(u)\Big) + \sum_{i=1}^{N}\sum_{j=1}^{M}\tau_{iq}(t)\eta_{j\ell}(t)\Big(1-\delta_{ij}(t)\Big)} \\ &= \frac{N_{q\ell}^{old} + N_{q\ell}^{(t)}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}} = \frac{N_{q\ell}^{old}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}} + \frac{N_{q\ell}^{(t)}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}}.\end{aligned} \tag{15}$$

By splitting Λ in two different parts, we can distinguish between a part known at time $t-1$, namely $N_{q\ell}^{old}$ and $D_{q\ell}^{old}$, and the current updates at time t, $N_{q\ell}^{(t)}$ and $D_{q\ell}^{(t)}$. Then, we divide and multiply the first term for $D_{q\ell}^{old}$, such that, denoting $\hat{\Lambda}_{q\ell}^{old} = \dfrac{N_{q\ell}^{old}}{D_{q\ell}^{old}}$, we obtain the final online update:

$$\hat{\Lambda}_{q\ell} = \hat{\Lambda}_{q\ell}^{old} \cdot \frac{D_{q\ell}^{old}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}} + \frac{N_{q\ell}^{(t)}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}}, \tag{16}$$

Hence, when a new observation comes Λ can be updated thanks to Eq.(16), along with the update of $D_{q\ell}^{old}$.

3.3.2 Update of α, β and π through deep neural networks

As mentioned in Section 2.1.3, the mixture proportions α and β, as well as the sparsity parameter π are driven by three systems of differential equations, respectively. Hence, the update process for these parameters in the online inference algorithm poses a challenge because they lack closed-form updating formulas. As a result, the decomposition strategy used for updating Λ cannot be applied. To address this issue, we introduce an approximation technique that leverages a moving window $G_d(t)$ of size d, allowing us to update the parameters based on the most recent d observations. In addition to its role in parameter updates, the moving window $G_d(t)$ serves another purpose as the input for a deep neural network. As we assumed that the functions f_A, f_W and f_Z are continuous, we propose to parametrize them with three LSTM networks (Hochreiter and Schmidhuber, 1997). LSTM is a type of recurrent neural network that operates on sequences of a specific length and produces a sequence of the same length, shifted one time step ahead. For instance, let's consider the current time t and the time window $G_d(t)$ with a length of d. The input for the LSTM networks consists of a series of values ranging from $t-1-d$ to $t-1$, representing the historical observations within the window. The LSTM networks then predict a sequence of values from $t-d$ to t, which correspond to the updated parameter values for α, β, and π. Optimizing the lower bound in Eq.(13) with respect to α, β, and π reduces to maximize it with respect to the parameters of the neural networks. For instance, the loss function for α can be expressed as follows:

$$\mathcal{L} = \sum_{u \in G_d(t)} \sum_{i=1}^{N} \sum_{q=1}^{Q} \tau_{iq}(u) \log \alpha_q(u), \tag{17}$$

The loss functions of β and π can be similarly derived using their respective distribution-specific equations. In the experiments, this update is implemented in PyTorch via automatic differentiation (Paszke et al., 2017) and relies on stochastic optimisation (ADAM, Kingma and Ba, 2014). Once the neural nets are trained via back-propagation (SGD) they provide us with the current ML estimates of $\alpha(t)$, $\beta(t)$ and $\pi(t)$. The whole inference procedure is summarized in Algorithm 1.

3.4 Initialization and model selection

When it comes to clustering methods based on the EM algorithm, the initialization process and selecting the appropriate number of clusters (for rows and columns in this case) are two important aspects that require careful consideration. These issues become slightly more complex when deep neural networks are employed to model cluster dynamics and sparsity proportions. Despite the inherent complexity introduced by these networks, they offer unexpected flexibility that can help reduce the computational burden of the entire algorithm. This becomes evident through the results of numerical experiments, as shown in Section 4.2.

In this section, modifications to the initialization method presented in Marchello et al. (2022a) are proposed to make the algorithm adaptable to run online. First, we select the first slice of the data X_{t_0} and apply on it a static LBM algorithm for a list of pairs of cluster numbers, i.e. (q, ℓ) for $q = 2, \ldots, Q_{max}$ and $\ell = 2, \ldots, L_{max}$. Subsequently, we employ the ICL criterion (Integrated Completed Likelihood, Biernacki et al. (2000)) to determine the optimal number of row and column clusters for this particular subset of data. The ICL criterion approximates the integrated log-likelihood of the complete dataset and can be derived as follows:

$$ICL(Q, L) = \log p(X, \hat{Z}, \hat{W}; \hat{\theta}) - \frac{Q-1}{2} \log N - \frac{L-1}{2} \log M - \frac{QL}{2} \log(NM) - \frac{1}{2} \log(NM). \tag{18}$$

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_0}. However, as we expect that the choice of $\hat{Q}$ row and $\hat{L}$ column cluster components could not be the best for all the future time instants, the VEM-SGD algorithm (see Algorithm 1) will be then run with more components than the ones found by the ICL. Indeed, we run the VEM-SGD algorithm with $Q_{max} \geq \hat{Q}$ and $L_{max} \geq \hat{L}$ cluster components. Then, every time there is a new data entry, part of the model parameters are initialized with $\hat{\theta}(t)$ obtained via a static run of LBM and the remaining parameters, corresponding to the additional row and column clusters are set to zero.

Therefore, our objective is to leverage the advantage of using deep neural networks, which enables our VEM-SGD algorithm to initialize with empty clusters. These empty clusters can potentially be activated in the future, if required. As a result, we can avoid the typically computationally intensive task of running the entire algorithm with all possible combinations of row and column cluster numbers for the complete dataset. This strategy enables our approach to handle large-scale datasets within a reasonable computation time while achieving satisfactory results, as demonstrated in the forthcoming section.
Also, at the initial stage of the algorithm (i.e. the first d time points), the parameters $\alpha(t)$, $\beta(t)$ and $\pi(t)$ are modeled via a two-layer fully connected neural networks, following the approach of Marchello et al. (2022a), until $t = d$. Then, as explained in Section 3.3.2, at $t = d + 1$, the estimates from the previous time step, obtained via fully connected networks, serve as input to LSTM, which is used for online parameter estimation from this point on.

Algorithm 1 VEM-SGD Algorithm for Stream Zip-dLBM

Require: $X, \hat{Q}, \hat{L}, Q_{max}, L_{max}, max.iter, G_d(t)$.

Initialization of $\tau(t=0)$ and $\eta(t=0)$: sampling from $\mathcal{M}(\alpha(t=0))$ and $\mathcal{M}(\beta(t=0))$, respectively;

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

while New observations $X(t)$ come: **do**

Initialization of $\alpha(t), \beta(t), \pi(t), \Lambda$ with LBM; % *with* $\hat{Q}$*, and* $\hat{L}$

▶ Add $Q_{max} - \hat{Q}$ columns of zeros to $\alpha(t)$;

▶ Add $L_{max} - \hat{L}$ columns of zeros to $\beta(t)$;

▶ Add $Q_{max} - \hat{Q}$ rows and $L_{max} - \hat{L}$ columns of zeros to Λ;

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

VE-Step:

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

alternatively update $\delta(t)$, $\tau(t)$, $\eta(t)$; % *fix point eqs*

end for

M-Step:

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

$$\hat{\Lambda}_{q\ell} = \hat{\Lambda}_{q\ell}^{old} \cdot \frac{D_{q\ell}^{old}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}} + \frac{N_{q\ell}^{(t)}}{D_{q\ell}^{old} + D_{q\ell}^{(t)}}.$$

Update $\alpha(t), \beta(t), \pi(t)$ %*LSTM on the moving window* $t \in G_d(t)$

end for

Discard all the observation before $G_d(t)$

end while

3.5 Bayesian online change point detection

As previously stated, one of the aims of Stream Zip-dLBM is to perform multivariate online change point detection. To accomplish this task, we combine the Bayesian Online Change Point Detection (BOCD) method, proposed in a seminal paper by Adams and MacKay (2007), with our strategy. BOCD detects change points based on the estimation of the posterior distribution over the current "run length", or time segment since the last change point, given the data observed so far, using a simple message-passing algorithm. Essentially, the run length is used to determine if a new data point belongs to the current partition based on previous observations. If the new data point belongs to the current partition, the run length will increase by 1 at the next time step, otherwise it will reset to 0. This process is continuously repeated at each time step. It is worth noticing that the BOCD algorithm is typically implemented in an online fashion, analyzing the data as it streams in. However, in our case, we directly apply the algorithm to detect change points on the estimates of $\alpha(t)$, $\beta(t)$, and $\pi(t)$ that are generated by the LSTM. To prevent detecting change points on parameters that will be recalculated in future time steps, we run the BOCD algorithm only on time points "behind" $G_d(t)$. Stated differently, at time t, BOCD operates on parameter values at time instances $t - d$.

4 Numerical experiments

The main purpose of this section is to highlight the most important features of Stream Zip-dLBM over simulated datasets and to demonstrate the validity of the inference algorithm presented in the previous sections. The first experiment consists in applying Stream Zip-dLBM to a specific dataset with evolving block pattern and sparsity to show that it recovers the data structure in real-time. While the second experiment demonstrates the model selection procedure on a simulated dataset.

4.1 Introductory example

A simulated dataset with dimension $350 \times 300 \times 150$ has been generated according to our model to perform this experiment. The simulated dynamics of α, β and π can be seen on the left-hand side of Figure 2. Concretely, α, β and π are three time series independently fluctuating around constant trends. Fluctuations are obtained at each time by means of independent Gaussian distributions with constant variance. The mean levels change when a change point occurs. The levels of the simulated change points and the values of the simulated parameter Λ are indicated in Table 1. Based on the mixture proportions α, β, the values of the latent variables were then simulated through their distributions. Next, we used the sparsity proportions, π, and the intensity parameter, Λ, to simulate the three-dimensional tensor X as Zero-Inflated Poisson variables. We then applied the Stream Zip-dLBM inference algorithm to the simulated dataset, using the actual values of $Q = 3$ and $L = 2$ to demonstrate the model's ability to recover the parameters. Figure 2 displays the true mixture proportions

α			
cp	16	66	103
β			
cp	25	62	112
π			
cp	16	66	103

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

Figure 1: Simulated time instants for change points of α, β and π and simulated values of Λ.

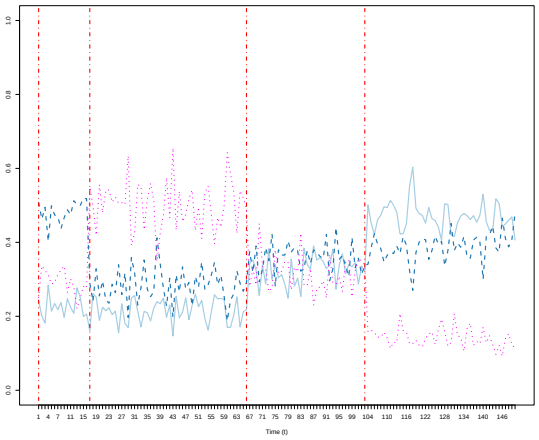
on the left side and the online estimates on the right side. The red dashed lines depict the simulated and estimated change points, respectively. From these results we see that Stream Zip-dLBM perfectly recovers the evolution of the mixing proportion and the sparsity parameter over time, including the change points.

4.2 Model selection experiment

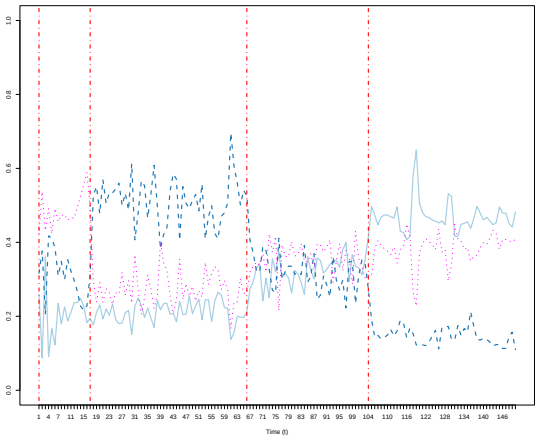
In this experiment, we assess the ability of Stream Zip-dLBM to determine the optimal number of clusters for rows and columns. Initially, we utilize the Integrated Completed Likelihood (ICL) criterion to compute the optimal number of clusters on the first slice. This ensures that the algorithm is initialized with the best possible parameters. Once initialized, the algorithm maintains consistent results without making any alterations, thus retaining the optimal number of clusters. To evaluate the effectiveness of the algorithm and verify if it activates new clusters, we generate a dataset based on the configuration described in Section 4.1. The dataset consists of 3 row clusters (Q=3) and 2 column clusters (L=2). We then apply Stream Zip-dLBM to this dataset, with maximum values of $Q_{max} = 7$ and $L_{max} = 7$. Figure 3 provides an illustrative demonstration of the algorithm's behavior, specifically regarding the activation of clusters. It is clear from the figure that the unnecessary clusters remain empty and that the estimates of the α, β, and π parameters are also accurate. Finally, it is worth noticing that Stream Zip-dLBM successfully identifies the changing points in α, β and π over time, despite not using the optimal number of input clusters. Finally, to evaluate the performance of the model in identifying the correct rows and columns partitions, we use the adjusted Rand index (ARI) (Rand, 1971). The adjusted Rand index, from a mathematical point of view, is closely related to the accuracy measure, however it is a commonly used method for evaluating clustering problems since it can be applied for measuring the similarity between two partitions even with different number of clusters and it is invariant to label switching. We also 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 these indexes are to 1, the more two label vectors are similar to each other. We compared the original matrices Z and W, with the estimates τ and η given by the output of Stream Zip-dLBM. We evaluate the performance indexes at each time step, obtaining the following results:

ARI rows	ARI columns	CARI
0.99 ± 0.03	1 ± 0	0.99 ± 0.02

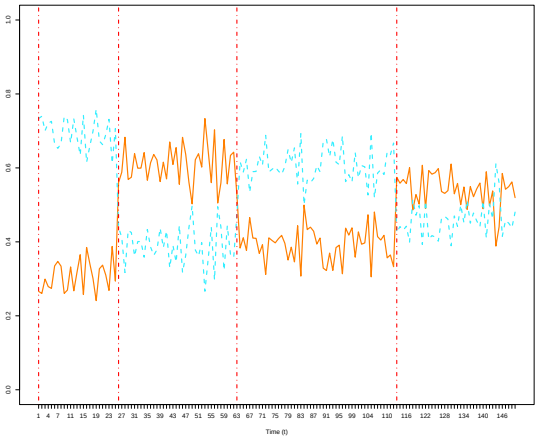
Thus, we can conclude that our algorithm satisfyingly identifies the composition of the original clusters in time.



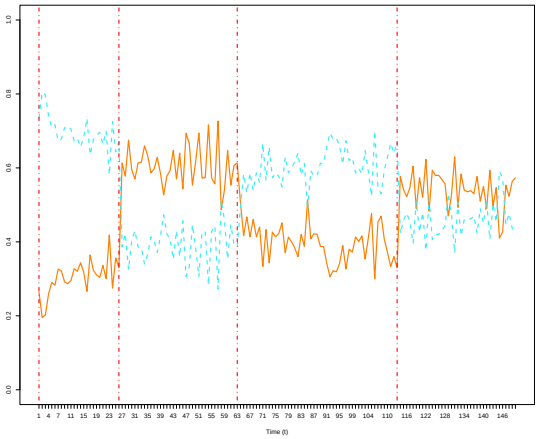
(a) True α.



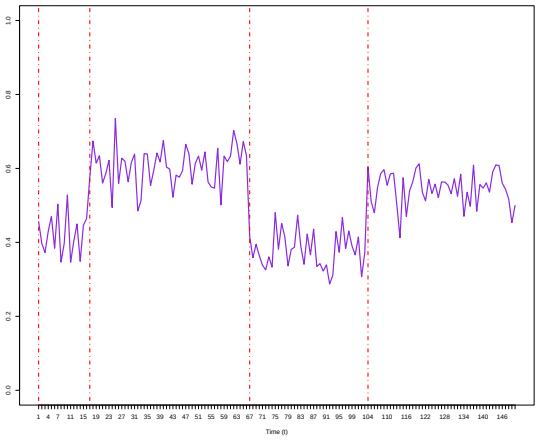
(b) Estimated α.



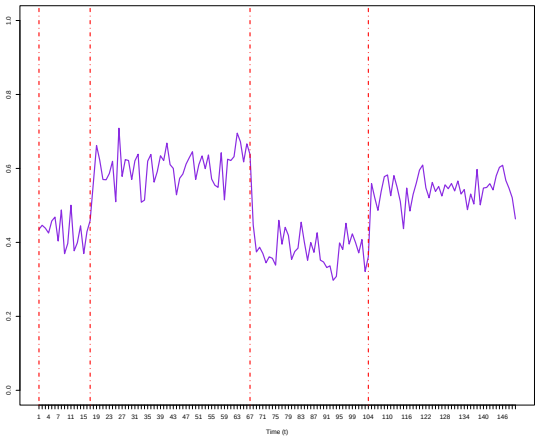
(c) True β.



(d) Estimated β.

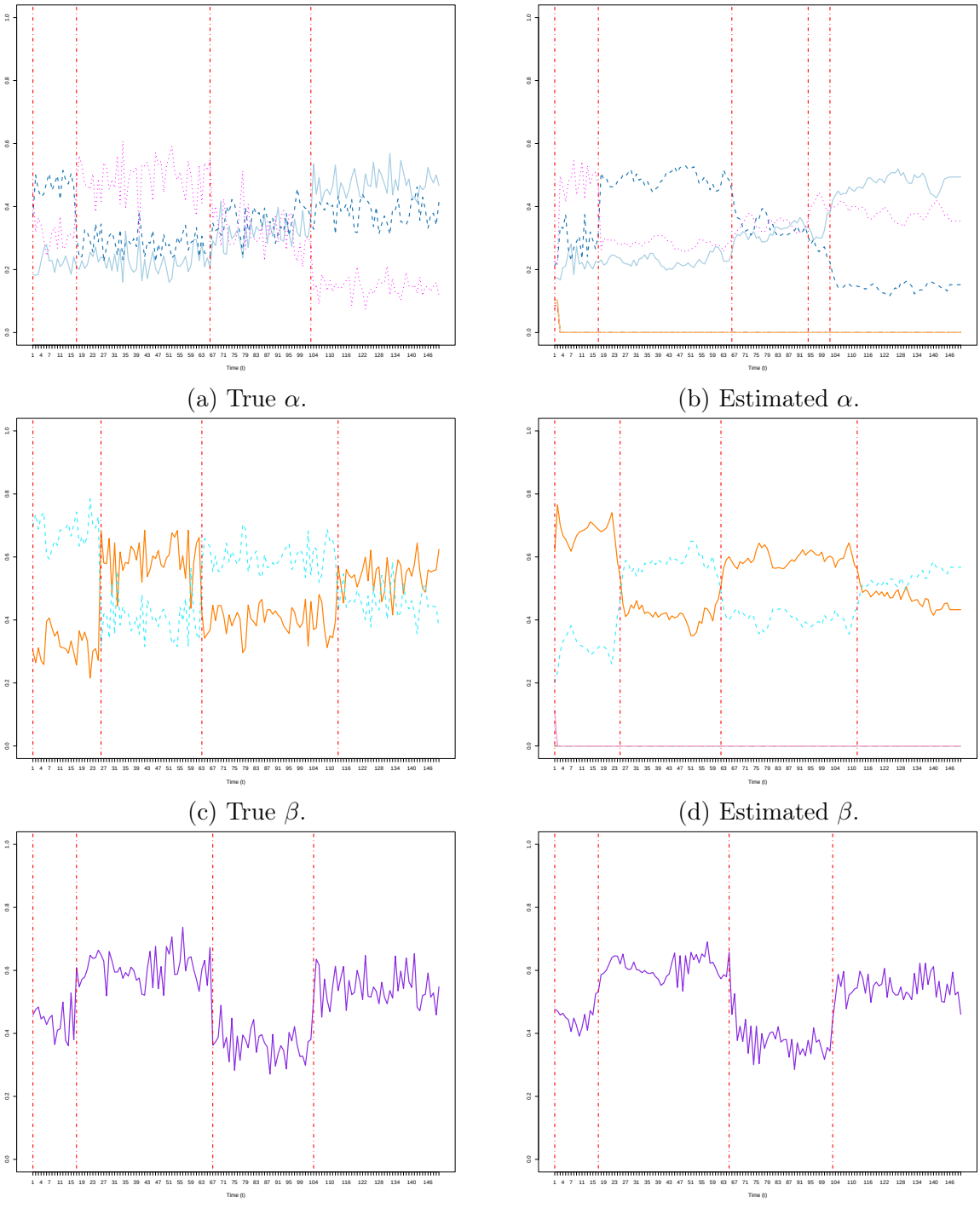


(e) True π.



(f) Estimated π.

Figure 2: Evolution of the true (left) and estimated (right) proportions of the parameters α, β and π, respectively.



(a) True α.

(b) Estimated α.

(c) True β.

(d) Estimated β.

(e) True π.

(f) Estimated π.

Figure 3: Evolution of the true (left) and estimated (right) proportions of the parameters α, β and π, respectively.

5 Analysis of the adverse drug reaction dataset

This section focuses on the online application of Stream Zip-dLBM to a large-scale pharmacovigilance dataset, with the aim of illustrating the potential of the tool for such studies.

5.1 Protocol and data

This section considers a large dataset consisting of an adverse drug reaction (ADR) dataset, collected by the Regional Center of Pharmacovigilance (RCPV), located in the University Hospital of Nice (France). The RCPV survey an area of three departments totalling 2.3 millions inhabitants. A time horizon of 7 years is considered, from January 1^{st}, 2015 to March 3^{rd}, 2022. Since the data are extremely sparse, we aggregate them summing up along the time dimension, such that one time instant corresponds to one month. The overall dataset is made of 39,267 declarations, for which the market name of the drug, the notified ADR and the reception date are considered. Moreover, we only considered drugs and ADRs that were notified more than 10 times over the 7 years. The resulting dataset contains 419 drugs, 614 ADRs and 87 time intervals with 23,264 non-zero entries. To prevent medicines with the same molecule but marketed under different brand names from being considered more than once, we decided to use the international nonproprietary name (INN) of the drug (to simplify the comprehension, the INNs would be referred as drugs for the rest of the study). Looking at Figure 4, it can be clearly noticed that there are at least two peaks to detect, one in 2017 and the other one in 2021.
In 2017, an unexpected rise of reports for ADRs happened concerning a specific drug called Lévothyrox®, containing the active compound levothyroxine. This has been marketed in France for about 40 years as a treatment for hypothyroidism and, in 2017, to correct drug stability problems, a new formula (with an innocuous change of excipients) was introduced on the market. The Lévothyrox® case had an extremely high media coverage in France: Lévothyrox® spontaneous reports represent almost the 90% of all the spontaneous notifications that the RCPV received in 2017 (Viard et al., 2019). In addition, since the end of the year 2020, vaccinations against Covid-19 have been introduced. At that time, three vaccines are licensed in Europe, Comirnaty® was the first Covid-19 vaccine available in France in December 2020, followed by Moderna® in January 2021 and Vaxzevria® in February 2021.

From Figure 4, one can understand the difficulty to work with such data which contain signals of very different amplitude. Indeed, behind those very visible effects, many other ADR signals, somehow less trivial, need to be detected for obvious public health reasons. In particular, those data also contain ADR reports regarding another public health problem which occurred in 2017, involving Mirena®, which is here far less visible than Lévothyrox®, but also led to many avoidable health policy concerns. This is why, we expect Stream Zip-dLBM to be a useful tool to reveal such hidden signals. It is important to highlight that the data being used exhibit extreme sparsity, ranging from a minimum of 99.25% to a maximum of 99.98% per month. To avoid encountering numerical issues, the LSTM network was employed only for inferring the parameters α and β. Point estimates of $\hat{\pi}$ were used in the inference process.

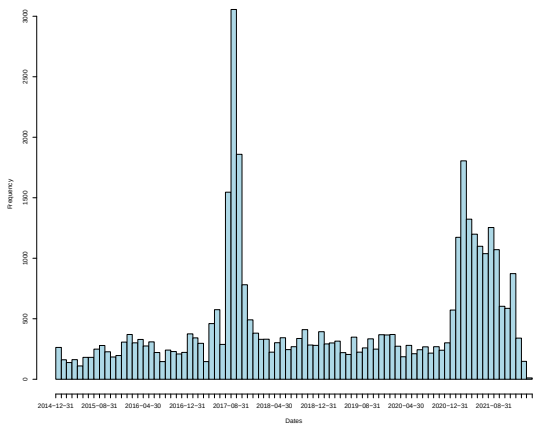


Figure 4: Number of declarations received by the pharmacovigilance center from 2015 to 2022, sorted by month.

5.2 Summary of the results

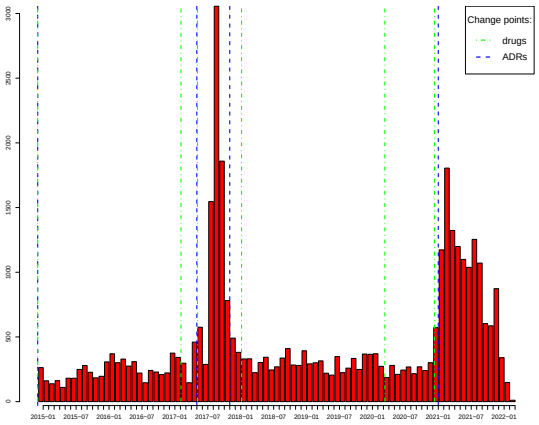
To initialize the algorithm, as explained in Section 3.4, we computed the ICL criterion on one data slice, corresponding to the first month, where the optimal numbers of clusters identified by the model selection criterion are $\hat{Q} = 3$ and $\hat{L} = 3$. Then, we run Stream Zip-dLBM and every time a new entry is added in the tensor X. We initiate the model parameters through the process described in Section 3.4. Also, we ran Stream Zip-dLBM with $Q_{max} = 7$ and $L_{max} = 7$ to allow the model to fill or empty clusters as needed. The process takes a running time of about one hour on a MacBook Pro, 2020, with a processor of 2,3 GHz Quad-Core Intel Core i7 and 16 GB of RAM.

In Figure 5a, the frequency of declarations received by the RCPV from 2015 to 2022 is depicted, organized by month. Here, the identified change points are represented by dashed lines. Specifically, the green dashed lines indicate the change points detected in the evolution of drug clusters, while the blue dashed lines indicate the change points detected in the evolution of ADR clusters.

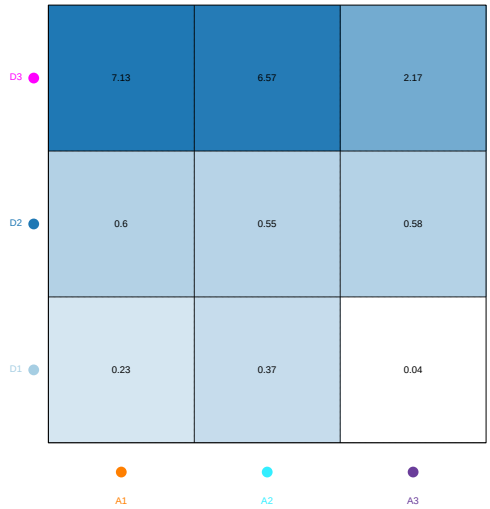
Figure 5b displays the estimated Poisson intensity parameter, Λ. This figure only focuses on the 3 groups of drug clusters, denoted by the letter D, and the 3 groups of ADR clusters, denoted by the letter A, that have been activated in the inference. This representation provides valuable insights for model interpretation as it gives an overview of the relationships between drug clusters and ADR clusters and how they evolve over time. Each color refers to a drug (rows) or ADR (columns) cluster and the higher is the value in each block, the strongest is the relationship (i.e the expected number of declarations received 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.

Figures 5c and 5d display the estimated mixture proportions of drug clusters ($\hat{\alpha}$) and ADR clusters ($\hat{\beta}$) respectively. The dashed lines in the figures represent the change points identified by the BOCD algorithm.

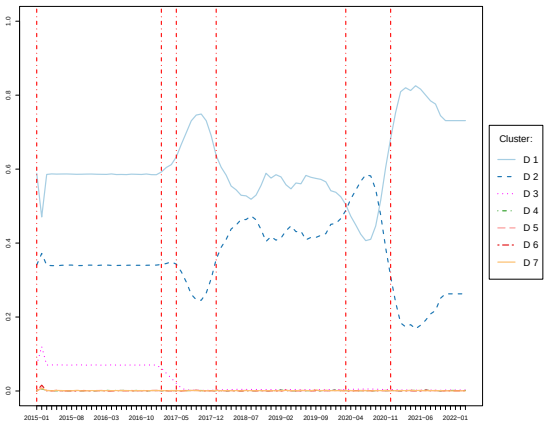
Figure 6 illustrates the estimated sparsity parameter, with the y-axis scale ranging from 0.99 to 1. It is worth noting that no change points have been detected in $\hat{\pi}$ because the values



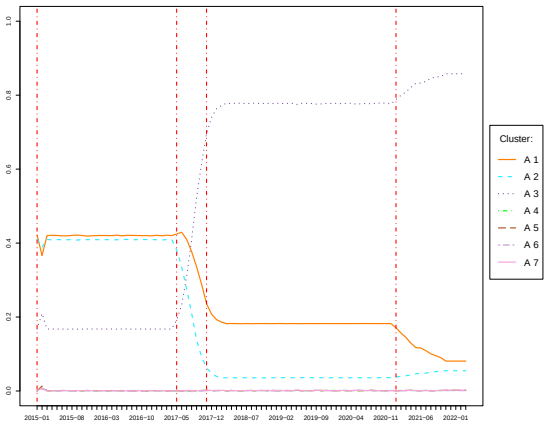
(a) Histogram of declarations over time, with change points.



(b) Estimated Poisson intensity parameter.



(c) Evolution of drug clusters proportions.



(d) Evolution of ADR clusters proportions.

Figure 5: Histogram of declarations over time, with the change points detected for the drugs and ADRs (top left) and estimated Poisson intensities, each color represents a different drug (ADR) cluster (top right); evolution of the estimates of $\hat{\alpha}$ (bottom left); evolution of the estimates of $\hat{\beta}$ (bottom right).

did not exhibit sufficient variability.

By examining the information provided in Figures 5b, 5c, and 5d, we can observe an interesting pattern. The clusters with the highest intensity are also the least populated. For instance, cluster D3 (drug cluster) demonstrates a remarkably high intensity of interactions with clusters A1 and A2 (ADR clusters). Despite its high intensity, cluster D3 appears to be relatively small in Figure 5c. This phenomenon is attributed to the presence of drugs associated with significant health crises that happened during the reporting period. Notably, Mirena® in the first half of 2017, Lévothyrox® in the latter part of 2017, and Covid-19

vaccines throughout 2021 were the primary drivers of these crises. Interestingly, each crisis period is marked by a detected change point in both the drug cluster proportions (Figure 5c) and ADR cluster proportions (Figure 5d).

Similarly, by analyzing the composition of clusters A1 and A2, it is possible to identify which ADRs were the most reported in each of the aforementioned crises. For instance, from the composition of cluster A2 we notice that the most reported side effects during the Mirena® health crisis are mostly hormonal ones, such as anxiety, heat shock, and aggressive behavior. Then, looking at Figure 5d, during the Lévothyrox® health crisis we notice a peak in the A3 cluster of adverse drug reactions, probably because the great media coverage that the scandal had in those years made people declare the most disparate side effects. Also, we see that in 2021 there is another peak, corresponding to the period of the Covid-19 vaccination. Here, the adverse drug reactions found in cluster A1 are mostly linked to problems related to the injection of the vaccine (e.g. arm pain, arm inflammation, skin reaction) or flu syndrome as a result. Cluster D2 presents an interesting contrast as it remains empty until August 2017. Subsequently, it contains a few but widely used drugs that are frequently reported, such as paracetamol, amoxicillin, and some popular antidepressants. The evolution of cluster D2's proportion, as shown in Figure 5c, aligns with the change points identified by the algorithm. This cluster begins to populate after the Lévothyrox® health crisis, with a peak in the early stages of the Covid-19 crisis, before the introduction of the vaccines. From Figure 5b, it can be observed that the intensity of interactions with clusters of undesirable effects does not differ significantly for cluster D2. Cluster D1, on the other hand, exhibits remarkably low interaction intensities and is densely populated by all other drugs. Initially, from the beginning of the analysis until 2017, it contains all drugs. However, after 2017, it includes only drugs with a low frequency of reports. From Figure 5c, we see that the evolution of Cluster D1 over time aligns with the change points identified by the algorithm.

Upon examining Figures 5b and 5d, a distinct behavior is observed in the clusters of adverse effects. Initially, until June 2017, cluster A1 contains all adverse effects. However, it gradually empties over time. Specifically, during the Mirena® crisis, only the adverse effects not associated with that particular health scandal remain in cluster A1. Subsequently, a significant change in the cluster membership occurs following the change point identified in October 2017. From this point onward, the number of ADRs in cluster A1 decreases significantly, and they are specifically related to Lévothyrox® (e.g. hair loss, cramps, insomnia, etc.). After the Lévothyrox® crisis, cluster A1 becomes empty until the subsequent change point detected in January 2021. From this moment until the peak of Covid-19 vaccine reports in February 2022, cluster A1 includes the main adverse effects reported for Covid-19 vaccines (e.g. pain at the vaccination site, skin rash, pericarditis, etc.).

Also, from Figure 5d we clearly notice as the Lévothyrox® crisis marked a cornerstone in the history of pharmacovigilance, probably because from this moment on people realized its importance and began to declare side effects of drugs and vaccines much more frequently.

Lastly, Figure 6 provides the estimated evolution of the sparsity parameter over time. The y-axis scale is set from 0.99 to 1 in order to visualize the changes over time. Notably, no change points were detected in the estimated parameter $\hat{\pi}$ due to insufficient variability in the values. Initially, in 2015, the sparsity is recorded at 99.83%. As we approach the peak in 2017, the number of declarations increases, leading to a decrease in sparsity, reaching a local minimum of 99.38%. Subsequently, as we approach the peak related to Covid-19 vaccines,

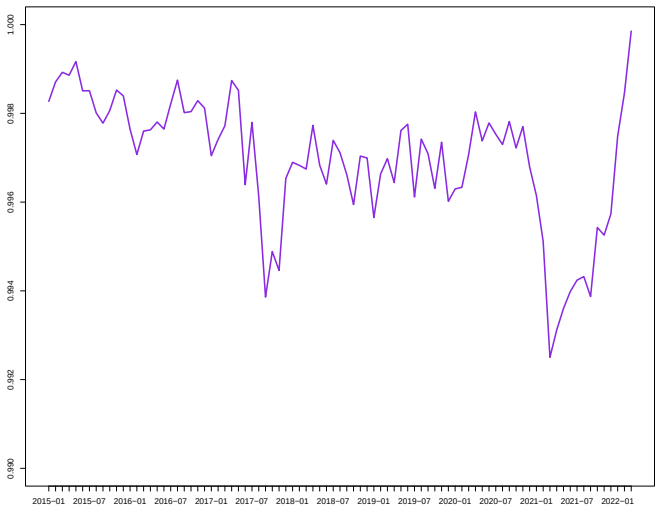


Figure 6: Evolution of the estimates $\hat{\pi}$.

the sparsity level reaches its global minimum at 99.25% in March 2021.
As a summary, Stream Zip-dLBM method successfully identified meaningful clusters within the extensive initial data matrix of ADR reports. This provides a proof of concept of the possible use of this algorithm to detect public safety events from streams of ADR data.

6 Conclusions

This paper is born out of the need to analyze and summarize observations and features of a dynamic matrix in an online setting for an application to pharmacovigilance. We have proposed an online dynamic co-clustering technique that enables simultaneous clustering of rows and columns along the time dimensions. As observations and features can change their cluster memberships over time, detecting structural changes in cluster interactions throughout the time period becomes crucial. We have introduced a generative zero-inflated dynamic latent block model as online extension of Zip-dLBM. The time modeling approach relies on three systems of ordinary differential equations. Inference is conducted using a Variational EM algorithm, combined with stochastic optimization of LSTM network parameters for the dynamic systems. Also, we added an online change points detection method to the process such that Stream Zip-dLBM is able to detect abrupt changes and create alerts in real time. The performance of our approach is evaluated through applications to some simulated data scenarios. Then, Stream Zip-dLBM was fit to a large-scale dataset supplied by the Regional Center of Pharmacovigilance of Nice (France). In this context, the model provided meaningful online segmentation of drugs and adverse drug reaction. Its potential use by medical authorities for identifying meaningful pharmacovigilance patterns or emerging public health concerns looks very promising. In fact, we are actively developing a web platform based on the Stream Zip-dLBM model for the Regional Center of Pharmacovigilance in Nice, France.

Once implemented, it will regularly run on a center machine, automatically fitting the model to incoming data. The goal is to promptly identify structural changes and send email notifications with concise reports. The software effectiveness will be assessed over 6 months, with plans to expand its use nationally. The online inference algorithm, combined with change point detection, allows Stream Zip-dLBM to operate in real-time, continuously analyzing the flow of ADR declarations and triggering alerts as soon as a change point is detected. This provides an opportunity for further investigation and intervention by medical authorities.

References

Adams, R. P. and MacKay, D. J. (2007). Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*.

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.

Basseville, M., Nikiforov, I. V., et al. (1993). *Detection of abrupt changes: theory and application*, volume 104. prentice Hall Englewood Cliffs.

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.

Cheng, S. W. and Thaga, K. (2005). Multivariate max-cusum chart. *Quality Technology & Quantitative Management*, 2(2):221–235.

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.

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.

Govaert, G. and Nadif, M. (2003). Clustering with block mixture models. *Pattern Recognition*, 36(2):463–473.

Govaert, G. and Nadif, M. (2010). Latent block model for contingency table. *Communications in Statistics - Theory and Methods*, 39(3):416–425.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jacques, J. and Biernacki, C. (2018). Model-based co-clustering for ordinal data. *Computational Statistics & Data Analysis*, 123:101–115.

Kavitha, V. and Punithavalli, M. (2010). Clustering time series data stream-a literature survey. *arXiv preprint arXiv:1005.4270*.

Kawahara, Y. and Sugiyama, M. (2009). Change-point detection in time-series data by direct density-ratio estimation. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 389–400. SIAM.

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*.

Kondratev, A., Salminen, K., Rantala, J., Salpavaara, T., Verho, J., Surakka, V., Lekkala, J., Vehkaoja, A., and Müller, P. (2022). A comparison of online methods for change point detection in ion-mobility spectrometry data. *Array*, page 100151.

Labiod, L. and Nadif, M. (2011). Co-clustering under nonnegative matrix tri-factorization. In *International Conference on Neural Information Processing*, pages 709–717. Springer.

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., Corneli, M., and Bouveyron, C. (2022a). A deep dynamic latent block model for the co-clustering of zero-inflated data matrices.

Marchello, G., Fresse, A., Corneli, M., and Bouveyron, C. (2022b). 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.

Montgomery, D. C. (2020). *Introduction to statistical quality control*. John Wiley & Sons.

Nowicki, K. and Snijders, T. A. B. (2001). Estimation and prediction for stochastic blockstructures. *Journal of the American statistical association*, 96(455):1077–1087.

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.

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.

Van den Burg, G. J. and Williams, C. K. (2020). An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*.

Viard, D., Parassol-Girard, N., Romani, S., Van Obberghen, E., Rocher, F., Berriri, S., and Drici, M.-D. (2019). Spontaneous adverse event notifications by patients subsequent to the marketing of a new formulation of levothyrox® amidst a drug media crisis: atypical profile as compared with other drugs. *Fundamental & clinical pharmacology*, 33(4):463–470.

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.