POLITECNICO
MILANO 1863

On a bayesian change point detection model for multivariate data

Chiara Bianchi, Lorenzo Dominoni, Piergiuseppe Pezzoli

Contents

1 Introduction and motivation **2**

2 Model **3**
- 2.1 Prior distribution of ρ_n . . . 3
- 2.2 Integrated Regime Likelihood . . . 4
- 2.3 Multivariate Ornstein-Uhlenbeck Process . . . 4
  - 2.3.1 Prior distribution . . . 5
  - 2.3.2 Posterior distribution . . . 5

3 MCMC simulation algorithm **7**
- 3.1 Details on the algorithm . . . 8

4 Performance on simulated data **10**
- 4.1 Changes in mean . . . 10
- 4.2 Changes in variance . . . 11
- 4.3 Changes in mean, variance and correlation . . . 12
- 4.4 Simulated data conclusions . . . 13

5 Real data applications **13**
- 5.1 Covid-19 cases in north, center and south Italy . . . 13
- 5.2 Closing prices of Milan stock exchange companies . . . 16
- 5.3 Real data conclusions . . . 20

6 Discussion **21**

References **21**

Supplementary Files **21**

Abstract

Change point detection models aim to determine the most probable grouping for a given sample indexed on an ordered set. We propose a procedure for detecting change points in multivariate data, based on exchangeable partition probability function. We will assume the Markovian property holds, in particular we will use discretely observed multivariate Ornstein-Uhlenbeck processes. We will explain some properties of the resulting model and we will use a custom Markov chain Monte Carlo method to obtain posterior results. We will validate this model on simulated data and we will test it on two real data applications: Covid-19 daily cases and finance stock prices.

1 Introduction and motivation

In this work we want to use a Bayesian approach to tackle multivariate time series. In particular we assume to see dependent data $y = (y_{t_1}, \dots, y_{t_n})$ which are observed at times $t_1, \dots, t_n$ with $0 < t_1 < \dots < t_n$. We try to identify changes in their underlying structure by focusing on the posterior distribution

$$\mathbb{P}(\rho_n|y) \propto \mathbb{P}(y|\rho_n)\mathbb{P}(\rho_n)$$

where ρ_n is a random variable modeling possible groupings for the data preserving the time ordering. The purpose is to model different behaviours of the phenomenon measured over time, in order to help the researcher to differentiate time periods (regimes) and identify notable events that separate those time periods. The application are several, and they cover a wide variety of fields: we can find examples in medical, financial, biological topics, as in several others. The time series paradigm well applies to modern science and experiments, and looking for change points can be crucial: we could be able to identify the moment when an epidemic began, when a stock price dropped, when a certain idea started spreading in a social network.

The inspiration of our work was the paper Martinez and Mena (2014) This is a paper which handles the study of time series using a Bayesian approach, but it has, in our view, a major limitation: it only covers one dimensional data. We think that looking to the evolution of a single phenomenon can be too reductive in our complex world: many events are strictly interlaced one with the other, and can influence a variety of changes in different ways across several time series. Taking into account only one feature at a time could leave out useful information to describe a problem. This is why we decided to work on a multivariate model.

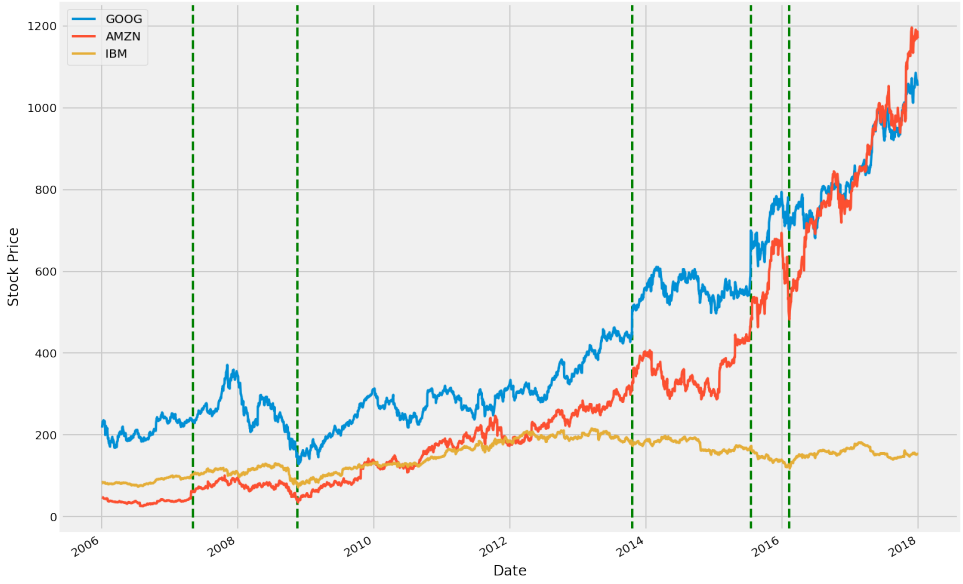

Figure 1: Stock Prices for three of the biggest companies in the world, Google, Amazon and IBM, from 2006 to 2018. We can see how there are points where there is a change in the behaviour.

2 Model

2.1 Prior distribution of ρ_n

We start by defining a random variable ρ_n which models all possible groupings for data preserving time ordering. Note that the set of all the partitions $\mathcal{P}_\mathcal{X}$ of an ordered set $\mathcal{X}$ and $\mathcal{P}_{[n]}$, where $[n] := 1, 2, ..., n$ are isomorphic combinatorial classes, than classifying $\mathcal{X}$ or $[n]$ is an equivalent problem. We use the latter to simplify the notations. We will use an Exchangeable Partition Probability Function (EPPF) to proceed with our work, since these classes of functions arise naturally with the Bayesian platform for exchangeable observations. An EPPF p is such that

$$\mathbb{P}(\Pi_n = A_1, ...A_k) = p(n_1, ..., n_k)$$

where $\{A_1, ...A_k\}$ is a partition of $[n]$ and $n_j := |A_j|$. EPPFs have been commonly used in many researches, but they need to be slightly modified for change point detection: we need to restrict their support over the set of the partitions that preserve the time ordering that is intrinsic in the data coming from a time series. We will follow the definition devised from Pitman (2006)

Definition 1. Let $\mathcal{C}_n$ be the set of order preserving partitions in $[n]$. A $\mathcal{C}_n - valued\, random\, variable\, \rho_n$ is said to have an exchangeable random order distribution if it is given by

$$p'(n_1, ..., n_k) = \binom{n}{n_1, ..., n_k} \frac{1}{k!} p(n_1, ..., n_k) \tag{1}$$

for p the EPPF of some random partition Π_n

An attractive statistic within the study and applications of EPPFs is the marginal distribution of K_n

$$\mathbb{P}(K_n = k) = \sum_{\{A_1, ...A_k\} \in \mathcal{P}_\mathcal{X}} \mathbb{P}(\Pi_n = A_1, ...A_k)$$

which indicates the number of partitions in a collection $\mathcal{X}$ of size n with exactly k groups. In our context, the random variable $\mathcal{C}_n = K_n - 1$ will be used to model the number of change points.

Here we will focus on the EPPF induced by the two-parameter Poisson-Dirichlet process, that is the EPPF characterized by the probability distribution

$$\mathbb{P}(\Pi_n = (n_1, ..., n_k)) = \frac{\prod_{i=1}^{k-1}(\theta + i\sigma)}{(\theta+1)_{n-1\uparrow}} \prod_{j=1}^{k} (1-\sigma)_{n_j - 1\uparrow} \tag{2}$$

where $(x)_{n\uparrow} = x(x+1)...(x+n-1)$ denotes the Pochhammer symbol, with $(x)_{0\uparrow} = 1$, and $\sigma \in [0, 1)$ with $\theta > -\sigma$ or $\sigma < 0$ with $\theta = m|\sigma|$ for some positive integer m. In particular we will work with the case $\sigma \in [0, 1)$ which, within the context at issue, corresponds to the case of always having new change points as the sample n increases.

Therefore, for this process, the restriction to compositions defined as in (1) simplifies to

$$\mathbb{P}(\rho_n = (n_1, ..., n_k)) = \frac{n!}{k!} \frac{\prod_{i=1}^{k-1}(\theta + i\sigma)}{(\theta+1)_{n-1\uparrow}} \prod_{j=1}^{k} \frac{(1-\sigma)_{n_j - 1\uparrow}}{n_j!} \tag{3}$$

In both cases, the marginal distribution for K_n is given by

$$\mathbb{P}(K_n = k) = \frac{\prod_{i=1}^{k-1}(\theta + i\sigma)}{\sigma^k (\theta+1)_{n-1\uparrow}} \frac{1}{k!} \sum_{j=0}^{k} (-1)^j \binom{k}{j} (-j\sigma)_{n\uparrow}, \qquad k = 1, ..., n$$

with mean value given by

$$\mathbb{E}[K_n] = \frac{(\theta+\sigma)_{n\uparrow}}{\sigma(\theta+1)_{n-1\uparrow}} - \frac{\theta}{\sigma} \tag{4}$$

This model, also known as the Pitman-Yor process, has been widely used in many fields.

2.2 Integrated Regime Likelihood

We impose dependency at the level of observations by assuming a continuous time Markovian process modulating each regime. Aferwards, we integrate out the process's driven parameters but we preserve the dependence induced through the correlation parameter. By doing this, we can focus on change points' inferences.

In particular, we assume data $y = (y_{t_1}, ..., y_{t_n})$ are modeled by a stationary Markovian process with invariant distribution $\pi(\cdot; x)$ and transition density $p(y_0, y_t; x)$, denoting generically the driven parameter by x. Then, the integrated regime likelihood is given by

$$\mathbb{P}(y|\rho_n = \{A_1, ..., A_k\}) = \prod_{j=1}^{k} L(y|A_j) \tag{5}$$

where $L(y|A_j)$ is the marginal likelihood given only the observations in group A_j, integrating out the parameter x, i.e.

$$L(y|A_j) = \int_{\mathbb{X}} \pi(y_{j,1}; x) \prod_{l=1}^{n_j - 1} p(y_{j,l}, y_{j,l+1}; x) P(dx) \tag{6}$$

where $y_{j,i}$ is the ith observation in group A_j, $n_j = |A_j|$ and P is the prior distribution of x. So, the integrated regime likelihood (5) is given by

$$\mathbb{P}(y|\rho_n = \{A_1, ..., A_k\}) = \prod_{j=1}^{k} \int_{\mathbb{X}} \pi(y_{j,1}; , x) \prod_{l=1}^{n_j - 1} p(y_{j,l}, y_{j,l+1}; x) P(dx) \tag{7}$$

2.3 Multivariate Ornstein-Uhlenbeck Process

We have been following so far the same procedure as in Martinez and Mena (2014). They continue the computations assuming a univariate Ornstein-Uhlenbeck process (Uhlenbeck and Ornstein (1930)), which is a well behaving stationary, reversible, Markovian and Gaussian process. To simplify things, they work under the assumption that observation are recorded at equally spaced times, thus using the simpler notation $t_i = i$ for $i = 1, ..., n$. This reduces the process to a $AR(1)$.

We think, as briefly explained in the first section, that working with multi-dimensional data instead of one-dimensional could give us a better model, capable of a great generalization. We decided to keep many of the assumptions, mainly the one regarding equally spaced times, that proved to be satisfied in many practical cases. So we propose a multivariate extension of the Ornstein-Uhlenbeck process. In literature is possible to find many examples of that, in particular we develop from the work of Vatiwutipong and Phewchean (2019).

Definition 2. Let $\{X_t\}$ be a multivariate sequence of random variables, with $X_t \in \mathbb{R}^d \, \forall t$. A multivariate Ornstein-Uhlenbeck process has a d-dimensional normal distribution with the following form:

$$X_{t+s}|X_s = x_s \sim \mathcal{N}_d(\mu + e^{B_t(x_s - \mu)}, \Lambda - e^{B_t} \Lambda e^{B_t^T}) \tag{8}$$

with equilibrium distribution:

$$X_t \sim \mathcal{N}_d(\mu, \Lambda)$$

We need to make first an isotropic assumption for the process, which consists in assuming that:

$$B = bI_d$$

where I_d is the identity matrix and $b < 0$.

Then we reparametrize in the following way:

$$\Phi = \Lambda - \Gamma \Lambda \Gamma^T$$

where $\Gamma = e^{B_t}$. In our case $t = 1$, since we have one observation for every time period. According to this assumption the vector of observations is the same in every direction, this simplifies the formulas and allows to handle more easily the problem.

$$\Gamma = I_d e^b = I_d \gamma$$
$$\Phi = \Lambda(1 - \gamma^2)$$

So we have a distribution of interest, which is the d-dimensional normal distribution just outlined

$$\begin{aligned} \underline{Y}_{t+1}|\underline{Y}_t = \underline{y}_t \sim \mathcal{N}_d(\underline{\mu} + \gamma(\underline{y}_t - \underline{\mu}); \Lambda(1-\gamma^2)) \\ \underline{Y}_t \sim \mathcal{N}_d(\underline{\mu}, \Lambda) \end{aligned} \tag{9}$$

where $\underline{y}, \underline{\mu} \in \mathbb{R}^d$, $\gamma \in (0,1)$ and $\Lambda \in \mathcal{M}_{d\times d}$

2.3.1 Prior distribution

We also need a prior distribution for $(\underline{\mu}, \Lambda)$ that we choose to be a Normal Inverse-Wishart, which is the conjugate prior of a Multivariate Normal distribution.

$$\begin{aligned} (\underline{\mu}, \Lambda) \sim NIW(m_0, k_0, \nu_0, \Psi_0) \\ \Rightarrow \begin{cases} \underline{\mu}|\Lambda \sim \mathcal{N}(m_0, \frac{\Lambda}{k_0}) \\ \Lambda \sim IW(\nu_0, \Psi_0) \end{cases} \end{aligned}$$

where $m_0 \in \mathbb{R}^d$, $k_0 > 0 \in \mathbb{R}$, $\Psi_0 \in \mathbb{R}^{d\times d}$ and $\nu_0 > d-1$

2.3.2 Posterior distribution

As introduced before, we will use a particular type of likelihood, i.e. the *integrated regime likelihood*, in order to infer the best partition without the concern of updating at each step the values of μ and λ. This step, although not essential, speeds up the algorithm and allows to focus just on the parameter we are interested in: the partition.

First of all we take the prior distribution and the likelihood:

$$\begin{aligned} \pi(\theta) = \pi(\underline{\mu},) \sim NIW(m_0, k_0, \nu_0, \Psi_0) \\ L(\underline{y}_{1:n}|\theta) = P(y_1|\theta) \prod_{i=2}^{n} P(y_i|y_{i-1}, \theta) \end{aligned}$$

The posterior distribution is proportional to the product between the prior distribution and the likelihood

$$\pi(\theta|\underline{y}_{1:n}) \propto \pi(\theta) L(\underline{y}_{1:n}|\theta)$$

which leads to

$$\begin{aligned} \pi(\theta|\underline{y}_{1:n}) = (2\pi)^{-\frac{d}{2}} (2\pi)^{-\frac{nd}{2}} \left|\frac{\Lambda}{k_0}\right|^{-\frac{1}{2}} \frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0 d}{2}} \Gamma_d(\frac{\nu_0}{2})} \frac{|\Lambda|^{\frac{-(\nu_0+n+d+1)}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}} \\ exp\left\{ -\frac{1}{2}(\mu - m_n)^T (\frac{\Lambda}{k_n})^{-1} (\mu - m_n) - \frac{1}{2} tr(\Psi_n \Lambda^{-1}) \right\} \end{aligned} \tag{10}$$

$$\begin{aligned} &\Gamma_d(a) = \pi^{\frac{d(d-1)}{4}} \prod_{j=1}^{d} \Gamma\left(a + \frac{1-j}{2}\right) \\ &\nu_n = \nu_0 + n \\ &k_n = \left[k_0 + \frac{(1-\gamma)^2}{(1-\gamma^2)}(n-1) + 1\right] \\ &m_n = \frac{1}{k_n}\left[m_0 k_0 + y_1 + \frac{(1-\gamma)}{(1-\gamma^2)} \sum_{i=2}^{n} (y_i - \gamma y_{i-1})\right] \\ &\Psi_n = \left(\psi_0 + y_1 y_1^\top + \sum_{i=2}^{n} \frac{(y_i - \gamma y_{i-1})(y_i - \gamma y_{i-1})^\top}{(1-\gamma^2)} + k_0(m_0 m_0^\top) + k_n(m_n m_n^\top)\right) \end{aligned}$$

Integrating out μ and Λ we obtain the formula that will be implemented inside the algorithm.

$$\mathbb{P}(\underline{y}|\rho_n) = \prod_{j=1}^{k} \frac{k_n^{-\frac{d}{2}}}{k_0^{-\frac{d}{2}}} \frac{(\pi)^{-\frac{nd}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}} \frac{|\Psi_0|^{\frac{\nu_0}{2}}}{\Gamma_d(\frac{\nu_0}{2})} \frac{\Gamma_d(\frac{\nu_n}{2})}{|\Psi_n|^{\frac{\nu_n}{2}}} \tag{11}$$

Proofs for (10) and (11) are out lined, respectively in (12) and (13).

Proof.

$$\pi(\theta)L(\underline{y}_{1:n}|\theta) = (2\pi)^{-\frac{d}{2}}\left|\frac{\Lambda}{k_0}\right|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(\mu-m_0)^T(\frac{\Lambda}{k_0})^{-1}(\mu-m_0)\right\}$$
$$\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0 d}{2}}\Gamma_d(\frac{\nu_0}{2})}|\Lambda|^{-\frac{(\nu_0+d+1)}{2}} exp\left\{-\frac{1}{2}tr(\Psi_0\Lambda^{-1})\right\}(2\pi)^{-\frac{d}{2}}|\Lambda|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(y_1-\mu)^T\Lambda^{-1}(y_1-\mu)\right\}$$
$$\prod_{i=2}^{n}(2\pi)^{-\frac{d}{2}}\left|\Lambda(1-\gamma^2)\right|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(y_i-\gamma y_{i-1}-(1-\gamma)\mu)^T\frac{\Lambda^{-1}}{(1-\gamma^2)}(y_i-\gamma y_{i-1}-(1-\gamma)\mu)\right\}$$
$$= (2\pi)^{-\frac{d}{2}}(2\pi)^{-\frac{nd}{2}}\left|\frac{\Lambda}{k_0}\right|^{-\frac{1}{2}}\frac{|\Lambda|^{-\frac{n}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0 d}{2}}\Gamma_d(\frac{\nu_0}{2})}|\Lambda|^{-\frac{(\nu_0+d+1)}{2}}$$
$$exp\left\{-\frac{1}{2}\left[\mu^T k_0\Lambda^{-1}\mu + m_0^T k_0\Lambda^{-1}m_0 - 2\mu^T k_0\Lambda^{-1}m_0 + tr(\Psi_0\Lambda^{-1}) + y_1^T\Lambda^{-1}y_1 + \mu^T\Lambda^{-1}\mu - 2\mu^T\Lambda^{-1}y_1 + \right.\right.$$
$$\left.\left. + \sum_{i=2}^{n}(y_i-\gamma y_{i-1})^T\frac{\Lambda^{-1}}{(1-\gamma^2)}(y_i-\gamma y_{i-1}) + (n-1)(1-\gamma)^2\mu^T\frac{\Lambda^{-1}}{(1-\gamma^2)}\mu - 2\mu^T\Lambda^{-1}\frac{(1-\gamma)}{(1-\gamma^2)}\sum_{i=2}^{n}(y_i-\gamma y_{i-1})\right]\right\}$$
$$= (2\pi)^{-\frac{d}{2}}(2\pi)^{-\frac{nd}{2}}\left|\frac{\Lambda}{k_0}\right|^{-\frac{1}{2}}\frac{|\Lambda|^{-\frac{n}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0 d}{2}}\Gamma_d(\frac{\nu_0}{2})}|\Lambda|^{-\frac{(\nu_0+d+1)}{2}}$$
$$exp\left\{-\frac{1}{2}\left[\mu^T\Lambda^{-1}\mu\left(k_0 + \frac{(1-\gamma)^2}{(1-\gamma^2)}(n-1)+1\right) - 2\mu^T\Lambda^{-1}\frac{k_n}{k_n}\left(k_0m_0 + y_1 + \frac{(1-\gamma)}{(1-\gamma^2)}\sum_{i=2}^{n}(y_i-\gamma y_{i-1})\right) + \right.\right.$$
$$+ m_n^T(\frac{\Lambda}{k_n})^{-1}m_n + tr(\Psi_0\Lambda^{-1}) + tr(y_1y_1^T\Lambda^{-1}) + tr\left(\sum_{i=2}^{n}(y_i-\gamma y_{i-1})(y_i-\gamma y_{i-1})^T\frac{\Lambda^{-1}}{(1-\gamma^2)}\right) +$$
$$\left.\left. + tr(k_0m_0m_0^T\Lambda^{-1}) - tr(k_nm_nm_n^T\Lambda^{-1})\right]\right\} =$$
$$= (2\pi)^{-\frac{d}{2}}(2\pi)^{-\frac{nd}{2}}\left|\frac{\Lambda}{k_0}\right|^{-\frac{1}{2}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0 d}{2}}\Gamma_d(\frac{\nu_0}{2})}\frac{|\Lambda|^{\frac{-(\nu_0+n+d+1)}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}exp\left\{-\frac{1}{2}(\mu-m_n)^T(\frac{\Lambda}{k_n})^{-1}(\mu-m_n) - \frac{1}{2}tr(\Psi_n\Lambda^{-1})\right\} \tag{12}$$

□

Proof.

$$\int_{S_\mu}\int_{S_\Lambda}(2\pi)^{-\frac{d}{2}}\frac{1}{k_0^{-\frac{d}{2}}}|\Lambda|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(\mu-m_n)^T(\frac{\Lambda}{k_n})^{-1}(\mu-m_n)\right\}$$
$$\frac{(\pi)^{-\frac{nd}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{\Gamma_d(\frac{\nu_0}{2})}\frac{|\Lambda|^{\frac{-(\nu_0+n+d+1)}{2}}}{2^{\frac{(\nu_0+n)d}{2}}}exp\left\{-\frac{1}{2}tr(\Psi_n\Lambda^{-1})\right\}d\mu d\Lambda =$$
$$= \frac{k_n^{-\frac{d}{2}}}{k_0^{-\frac{d}{2}}}\frac{(\pi)^{-\frac{nd}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{\Gamma_d(\frac{\nu_0}{2})}\frac{\Gamma_d(\frac{\nu_n}{2})}{|\Psi_n|^{\frac{\nu_n}{2}}}\int_{S_\mu}\int_{S_\Lambda}(2\pi)^{-\frac{d}{2}}\left|\frac{\Lambda}{k_n}\right|^{-\frac{1}{2}} exp\left\{-\frac{1}{2}(\mu-m_n)^T(\frac{\Lambda}{k_n})^{-1}(\mu-m_n)\right\} \tag{13}$$
$$\frac{|\Lambda|^{\frac{-(\nu_n+d+1)}{2}}}{2^{\frac{\nu_n d}{2}}}\frac{|\Psi_n|^{\frac{\nu_n}{2}}}{\Gamma_d(\frac{\nu_n}{2})}exp\left\{-\frac{1}{2}tr(\Psi_n\Lambda^{-1})\right\}d\mu d\Lambda =$$
$$= \frac{k_n^{-\frac{d}{2}}}{k_0^{-\frac{d}{2}}}\frac{(\pi)^{-\frac{nd}{2}}}{(1-\gamma^2)^{\frac{(n-1)}{2}}}\frac{|\Psi_0|^{\frac{\nu_0}{2}}}{\Gamma_d(\frac{\nu_0}{2})}\frac{\Gamma_d(\frac{\nu_n}{2})}{|\Psi_n|^{\frac{\nu_n}{2}}}$$

□

3 MCMC simulation algorithm

Even when the number of set compositions is much smaller than the number of partitions, simulating from the posterior distribution of ρ_n

$$\mathbb{P}(\rho_n|y) \propto \left(\frac{n!}{k!} \frac{\prod_{i=1}^{k-1}(\theta + i\sigma)}{(\theta+1)_{n-1\uparrow}} \prod_{j=1}^{k} \frac{(1-\sigma)_{n_j-1\uparrow}}{n_j!} \right) \mathbb{P}(y|\rho_n) \tag{14}$$

where $\mathbb{P}(y|\rho_n)$ is given by (11) becomes unfeasible for bigger datasets. The need of an MCMC algorithm is then evident. We use a split-merge algorithm as in Fuentes–García, Mena, and Walker (2010) which updates the number of groups k and the group sizes $(n_1, ..., n_k)$ using a Metropolis-Hastings step.

Hence two possible choices are available at each step: a split, which creates a new group, or a merge, which combines two consecutive existing groups into a single one. After that, a random pair of adjacent groups is updated proposing new values in order to speed the sampler up, in an operation named "shuffle".

Additionally, we assign hyperprior distribution to parameters θ, σ and γ so that we are also able to make inference about them. For the variables (σ, θ), which are the parameters related to the prior on the partitions the likelihood function is taken from the prior of ρ_n in (3), and we assume the following hyperpriors:

$$\sigma \sim Beta(a, b)$$
$$\theta|\sigma \sim ShiftedGamma(c, d, -\sigma)$$

For γ, which controls the correlation of the time series, we take the likelihood from $\mathbb{P}(\underline{y}|\rho_n)$, and we assume a uniform prior:

$$\gamma \sim Unif(0, 1)$$

We simulate at each step these parameters using an adaptive Metropolis rejection sampling (ARMS) and we update them at each iteration. This allows the algorithm to be more flexible, to better explore partitions without the need to spend much time tuning parameters.

The final algorithm is synthesized in the following table:

Algorithm 1: Split-Merge MCMC scheme

read $y_1, ..., y_n$ and hyper-parameters ;
initialize k and $(n_1, ..., n_k)$;
for N_{rep} **do**
  sample $U, V, T \overset{i.i.d.}{\sim} Unif(0, 1)$;
  if $U < q\mathbb{I}(1 < k < n) + \mathbb{I}(k = 1)$ **then**
    choose j uniformly from $\{j : 1 \leq j \leq k, n_j > 1\}$;
    choose l uniformly from $\{1, ..., n_j - 1\}$;
    if $V < \alpha$ *in* (15) **then**
      $(n_1, ..., n_{k+1}) \leftarrow (n_1, ..., n_{j-1}, l, n_j - l, n_{j+1}, ..., n_k)$;
      $k \leftarrow k + 1$;
    end
  else
    choose j uniformly from $\{1, ..., k - 1\}$;
    if $V < \alpha$ *in* (16) **then**
      $(n_1, ..., n_{k-1}) \leftarrow (n_1, ..., n_{j-1}, n_j + n_{j+1}, n_{j+2}, ..., n_k)$;
      $k \leftarrow k - 1$;
    end
  end
  if $k > 1$ **then**
    choose i uniformly from $\{1, ..., k - 1\}$;
    choose j uniformly from $\{1, ..., n_i + n_{i+1} - 1\}$;
    if $T < \alpha$ *in* (17) **then**
      $n_{i+1} \leftarrow n_i + n_{i+1} - j$;
      $n_i \leftarrow j$;
    end
  end
  simulate the additional parameters θ, σ and γ using ARMS;
end

3.1 Details on the algorithm

Let $p(k, \gamma_k) = p(n_1, ..., n_k|\underline{y})$ where γ_k is a grouping for the data $\underline{y}$ of size k which each group having size n_j.

As explained in details in Martinez and Mena (2014), we update k via Metropolis-Hastings with target distribution $p(k|\gamma)$, using the proposal distribution

$$p(k|r) = q\mathbb{I}(k = r + 1) + (1 - q)\mathbb{I}(k = r - 1),\ \ 0 < q < 1,\ 1 < r < n$$
$$p(2|1) = p(n - 1|n) = 1\ \ otherwise$$

we can compute the acceptance probability to update k, which is given by

$$\alpha = min\left\{\frac{p(k|k')}{p(k'|k)}\frac{p(k'|\gamma)}{p(k|\gamma)}, 1\right\}$$

with k' simulated from the proposal described before.

Since the nature of the split-merge MCMC algorithm limits the possible updates of k, we can further simplify.

For a split we have

$$\alpha = \begin{cases} min\left\{\frac{1-q}{q}\frac{p(k+1,\gamma_{k+1})}{p(k,\gamma_k)}\frac{n_{g,k}(n_s-1)}{k}, 1\right\}\ \ if\ \ 1 < k < n \\ min\left\{(1-q)(n-1)\frac{p(2,\gamma_2)}{p(1,\gamma_1)}, 1\right\}\ \ if\ \ k = 1 \end{cases} \tag{15}$$

where $n_{g,k}$ is the number of groups of size greater than one and n_s the size of the selected group.

For a merge we have

$$\alpha = \begin{cases} min\left\{\frac{q}{1-q}\frac{p(k-1,\gamma_{k-1})}{p(k,\gamma_k)}\frac{k-1}{n_{g,k-1}(n_s+n_{s+1}-1)}, 1\right\}\ \ if\ \ 1 < k < n \\ min\left\{q(n-1)\frac{p(n-1,\gamma_{n-1})}{p(n,\gamma_n)}, 1\right\}\ \ if\ \ k = n \end{cases} \tag{16}$$

where $n_{g,k-1}$ is the number of groups of size greater than one and n_s, n_{s+1} the sizes of the selected groups.

And for a shuffle we have

$$\alpha = min\left\{\frac{p(k, \gamma_k^*)}{p(k, \gamma_k)}, 1\right\} \tag{17}$$

where γ_k, γ_k^* are the partition respectively before and after the shuffle.

Moreover, we can reduce the computational load by inspecting in each case the ratios

$$\frac{p(\cdot, \cdot)}{p(*, *)}$$

that are present in every equation. Remember that this was a notation introduce to represent the posterior, so we can decompose them as a product of a ratio of priors and a ratio of likelihoods.

The ratio of the likelihoods simplifies as:

Split

$$\left(\frac{k_l k_{n_j-l}}{k_{n_j} k_0}\right)^{-\frac{d}{2}} \cdot \frac{1}{(1-\gamma^2)^{-\frac{1}{2}}} \cdot \frac{\Gamma_d(\frac{\nu_l}{2})\Gamma_d(\frac{\nu_{n_j-l}}{2})}{\Gamma_d(\frac{\nu_{n_j}}{2})\Gamma_d(\frac{\nu_0}{2})} \cdot \frac{|\psi_0|^{\frac{\nu_0}{2}}|\psi_{n_j}|^{\frac{\nu_{n_j}}{2}}}{|\psi_l|^{\frac{\nu_l}{2}}|\psi_{n_j-l}|^{\frac{\nu_{n_j-l}}{2}}}$$

Merge

$$\left(\frac{k_{n_j+n_{j+1}} k_0}{k_{n_j} k_{n_{j+1}}}\right)^{-\frac{d}{2}} \cdot \frac{1}{(1-\gamma^2)^{\frac{1}{2}}} \cdot \frac{\Gamma_d(\frac{\nu_{n_j+n_{j+1}}}{2})\Gamma_d(\frac{\nu_0}{2})}{\Gamma_d(\frac{\nu_{n_j}}{2})\Gamma_d(\frac{\nu_{n_j+1}}{2})} \cdot \frac{|\psi_{n_j}|^{\frac{\nu_{n_j}}{2}}|\psi_{n_{j+1}}|^{\frac{\nu_{n_j+1}}{2}}}{|\psi_{n_j+n_{j+1}}|^{\frac{\nu_{n_j+n_{j+1}}}{2}}|\psi_0|^{\frac{\nu_0}{2}}}$$

Shuffle

$$\left(\frac{k_j k_{n_i+n_{i+1}-j}}{k_{n_i} k_{n_{i+1}}}\right)^{-\frac{d}{2}} \cdot \frac{\Gamma_d(\frac{\nu_j}{2})\Gamma_d(\frac{\nu_{n_i+n_{i+1}-j}}{2})}{\Gamma_d(\frac{\nu_{n_i}}{2})\Gamma_d(\frac{\nu_{n_{i+1}}}{2})} \cdot \frac{|\psi_{n_i}|^{\frac{\nu_{n_i}}{2}}|\psi_{n_{i+1}}|^{\frac{\nu_{n_{i+1}}}{2}}}{|\psi_j|^{\frac{\nu_j}{2}}|\psi_{n_i+n_{i+1}-j}|^{\frac{\nu_{n_i+n_{i+1}-j}}{2}}}$$

Whereas the ratio of priors simplifies as:

Split

$$\begin{cases} \frac{(\theta+k\sigma)}{(k+1)}\binom{n_j}{l}\frac{(1-\sigma)_{n_j-l-1\uparrow}}{(l-\sigma)_{n_j-l\uparrow}} & \text{if } l \geq n_j - l \\ \frac{(\theta+k\sigma)}{(k+1)}\binom{n_j}{n_j-l}\frac{(1-\sigma)_{l-1\uparrow}}{(n_j-l-\sigma)_{l\uparrow}} & \text{otherwise} \end{cases}$$

Merge

$$\begin{cases} \frac{k}{(\theta+(k-1)\sigma)}\frac{1}{\binom{n_j+n_{j+1}}{n_j}}\frac{(n_j-\sigma)_{n_{j+1}\uparrow}}{(1-\sigma)_{n_{j+1}-1\uparrow}} & \text{if } n_j \geq n_{j+1} \\ \frac{k}{(\theta+(k-1)\sigma)}\frac{1}{\binom{n_j+n_{j+1}}{n_{j+1}}}\frac{(n_{j+1}-\sigma)_{n_j\uparrow}}{(1-\sigma)_{n_j-1\uparrow}} & \text{otherwise} \end{cases}$$

Shuffle

defining:

$$z_1 = \min\{\min\{n_i, n_{i+1}\}, \min\{j, n_i + n_{i+1} - j\}\}$$
$$z_2 = \max\{\min\{n_i, n_{i+1}\}, \min\{j, n_i + n_{i+1} - j\}\}$$
$$z_3 = \min\{\max\{n_i, n_{i+1}\}, \max\{j, n_i + n_{i+1} - j\}\}$$
$$z_4 = \max\{\max\{n_i, n_{i+1}\}, \max\{j, n_i + n_{i+1} - j\}\}$$

We get the simplification:

$$\begin{cases} \frac{(z_1+1)\cdots(z_3)}{(z_2+1)\cdots(z_4)}\frac{(z_2-\sigma)_{z_4-z_2\uparrow}}{(z_1-\sigma)_{z_4-z_2\uparrow}} & \text{if } \max\{n_i, n_{i+1}\} \geq \max\{j, n_i + n_{i+1} - j\} \\ [\frac{(z_1+1)\cdots(z_3)}{(z_2+1)\cdots(z_4)}\frac{(z_2-\sigma)_{z_4-z_2\uparrow}}{(z_1-\sigma)_{z_4-z_2\uparrow}}]^{-1} & \text{otherwise} \end{cases}$$

All the computations during the execution of the algorithm are done in log scale, because of the presence of huge numbers. In this way we overcome the computational issues and limit round off errors.

4 Performance on simulated data

To test the performances of our model and algorithm, we proceed by creating several simulated datasets where we know the true change points characterizing the multivariate time series andwe can compare them with the ones that we extract with our method.

Note that the Ornstein-Uhlenbeck process is characterized by three parameters: a mean vector $\underline{\mu}$, a covariance matrix Φ and a correlation scalar γ.

$$OU(\underline{\mu}, \Phi, \gamma)$$

We experimented with different changes for the process: at a given change point a change in mean, variance or both was applied, to investigate the performance of the algorithm in various scenarios.

In these simulations we present, we will keep the dimension of the data and the number of change points small to better help visualizing the problem and the results. Other trials were, however, made with more complex datasets, but we decided to show more complex behaviours using real-world data in the next section.

We used, during the tests, the following parameters initialization:
$a = 1, b = 1, c = 1, d = 1, \theta_0 = 0.1897, \sigma_0 = 0.1, m_0 = mean(\underline{y}), \nu_0 = 4, k_0 = 0.1, \Psi_0 = var(\underline{y}), \gamma_0 = 0.5, \underline{n}^{initial} = (30, 30, 30, 30), k^{initial} = 4, q = 0.5, N_{rep} = 20000, N_{burnin} = 5000$

4.1 Changes in mean

We simulate from the following process:

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}, 0.4\right), \quad i = 1, ..., 50$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}4\\4\\4\end{pmatrix}, \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}, 0.4\right), \quad i = 51, ..., 85$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}4\\4\\-4\end{pmatrix}, \begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}, 0.4\right), \quad i = 86, ..., 150$$

Here we have two change points, one for $t = 51$ and one for $t = 86$. The first introduces a change in the means of all the components, whereas the second modifies just one of the three means. All the other parameters remain unchanged during the simulation.

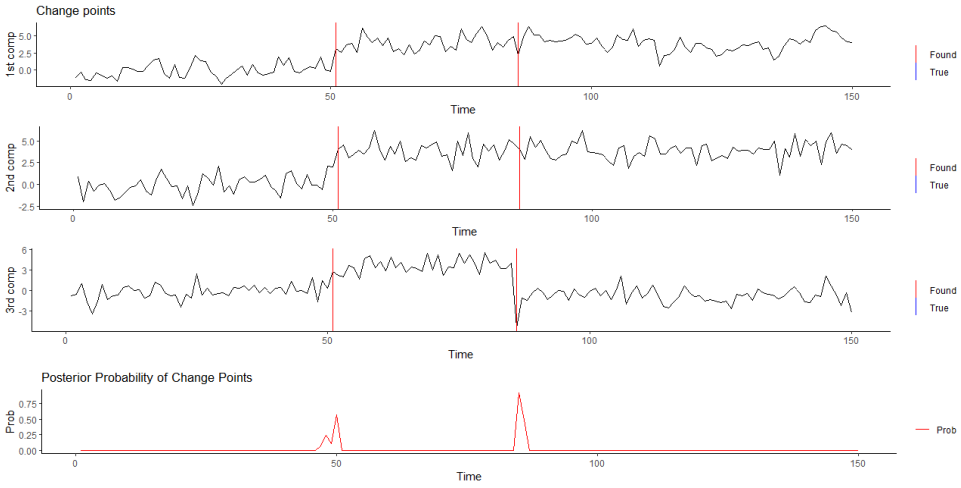

Figure 2: Realization of the simulated process, with the 3 components represented separately. In blue we highlighted the true change points and in red the one we produced with our algorithm. Here only the red lines are visible since they overlap with the blue ones. The fourth graph is the posterior probability for each point to be a change point.

From Figure 2 we can evaluate the performance of our algorithm: we have plotted each component of the time-series and we have highlighted with a blue line the true change points present, and with a

red line the change points we inferred. These values are extracted from the sequence of partitions that we explored with our split-merge MCMC algorithm using as estimate the mode of the sequence. This is a very simple statistic, but it proved to work well in simulated cases.

Note that our proposed values (red lines) coincide perfectly with the true values (blue lines), hiding them from the plot. We were very satisfied with the perfect performance of our algorithm in this case, but is probably the simplest one we could think of. Remember also that the number of change points is not decided a prior by the researcher, but is extracted from the MCMC simulation, so the approach is completely unsupervised in this choice, and accordingly more difficult. As estimate for the number of change points we suggest to choose k coherent with the number of change points in the mode, more interpretable with respect to the standard mode of k given by the algorithm.

An interesting result we can produce is the posterior probability for each t_{n_j} to be a change point. This is directly taken from the bayesian approach we are using, and allows us to quantify also the uncertainty of the inference, a factor that will be extremely useful when tackling more complex cases, as we are going to see. Moreover, the fact that the $t = 86$ point is detected reveals that suffices that only one component changes to find a change point.

The fourth plot that can be seen in Figure 2 is exactly this posterior probability that we explained. Note how in this easy case we have two spikes in exactly the true change points, which proves that the algorithm has no doubts when identifying them.

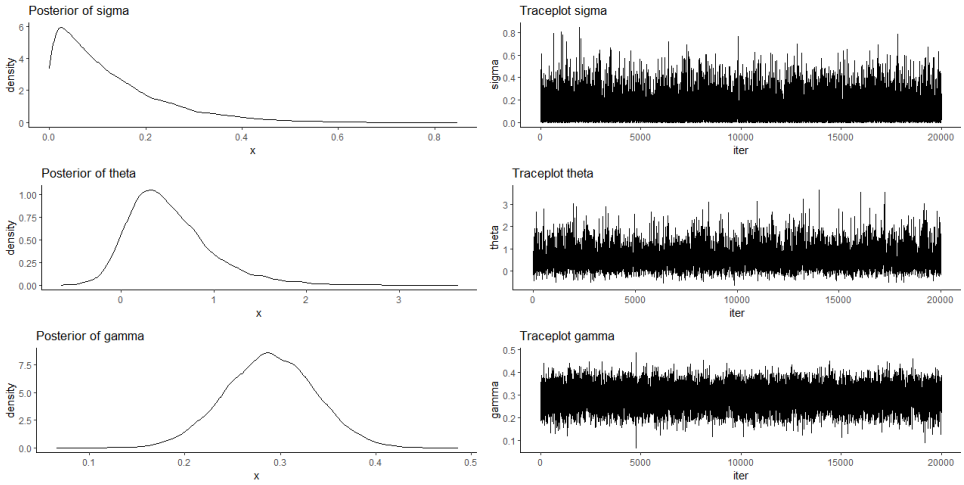

Figure 3: Posterior distribution and traceplots of the simulated additional parameters.

Figure 3 shows the posterior distributions and the traceplots of the parameters simulated by the ARMS method. Although we are not directly interested in them they are useful to check that the algorithm works correctly. In fact the posteriors are smooth and correctly placed around the real values, the traceplots don't show signs of correlation.

4.2 Changes in variance

The next simulation we do is from the following process:

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{pmatrix}, 0.4\right), \quad i = 1, ..., 50$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}5 & 0 & 0\\0 & 5 & 0\\0 & 0 & 5\end{pmatrix}, 0.4\right), \quad i = 51, ..., 85$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}5 & 4 & 4\\4 & 5 & 4\\4 & 4 & 5\end{pmatrix}, 0.4\right), \quad i = 86, ..., 150$$

Here we have two change points as before in the same spots $t = 51$ and $t = 86$. This time, however, we try to change only the variances and covariances of the process. The first change modifies indeed all the

variances, maintaining variables uncorrelated, whereas the second adds covariances. We keep mean and correlation fixed in order to inspect the accuracy of the model in this specific scenario.

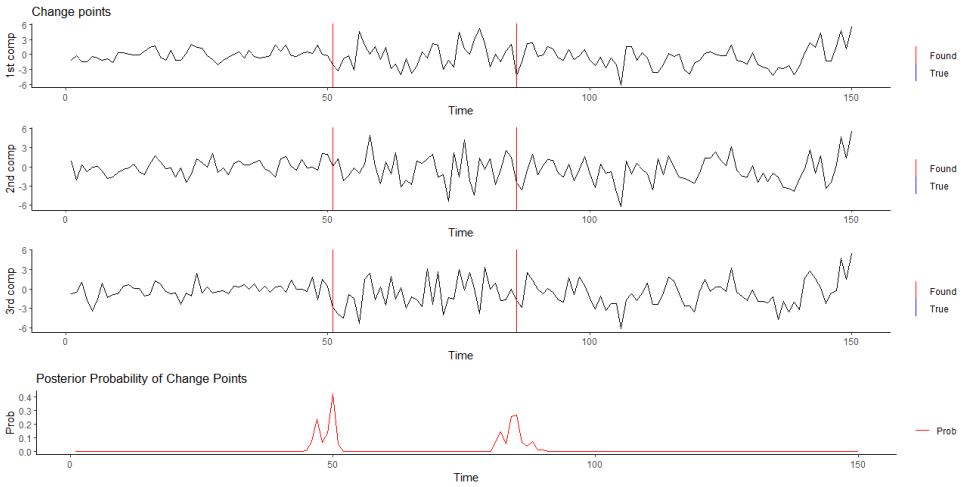

Figure 4: Realization of the simulated process, with the 3 components represented separately. In blue we highlighted the true change points and in red the one we produced with our algorithm. Here only the red lines are visible since they overlap with the blue ones. The fourth graph is the posterior probability for each point to be a change point.

From Figure 4 we can note that our estimated change points still coincide with the true ones: the lines highlighting them are overlapping as they previously did.

This time, however, we can see the importance of the posterior probabilities that we compute, as they give us a richer analysis of performance. The algorithm is still very capable of finding the right values that we are looking for, but we are warned that the probability is more "spread out", and thus there is more uncertainty on the underlying phenomena.

It's also interesting to notice that the change point in $t = 86$ would not be detected by a univariate approach, since it only involves covariances between variables. This motivates further our proposal with respect to its univariate counterpart.

4.3 Changes in mean, variance and correlation

The last simulation is sampled from this process:

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{pmatrix}, 0.4\right), \quad i = 1, ..., 50$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}3\\3\\3\end{pmatrix}, \begin{pmatrix}3 & 0 & 0\\0 & 3 & 0\\0 & 0 & 3\end{pmatrix}, 0.7\right), \quad i = 51, ..., 85$$

$$\underline{y}_i \sim OU\left(\begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}2 & 1 & 1\\1 & 2 & 1\\1 & 1 & 2\end{pmatrix}, 0.9\right), \quad i = 86, ..., 150$$

This last simulation was made with real data in mind. We tried to make the process as complex as we could, without changing the small number of change points presents (as before we have two: one at $t = 51$ and the other at $t = 86$). At each of those we modify mean, variance matrix and correlation. Notice that in our model we don't include the possibility for the correlation to change, so with this experiment we want also to test the robustness with respect to this hypothesis.

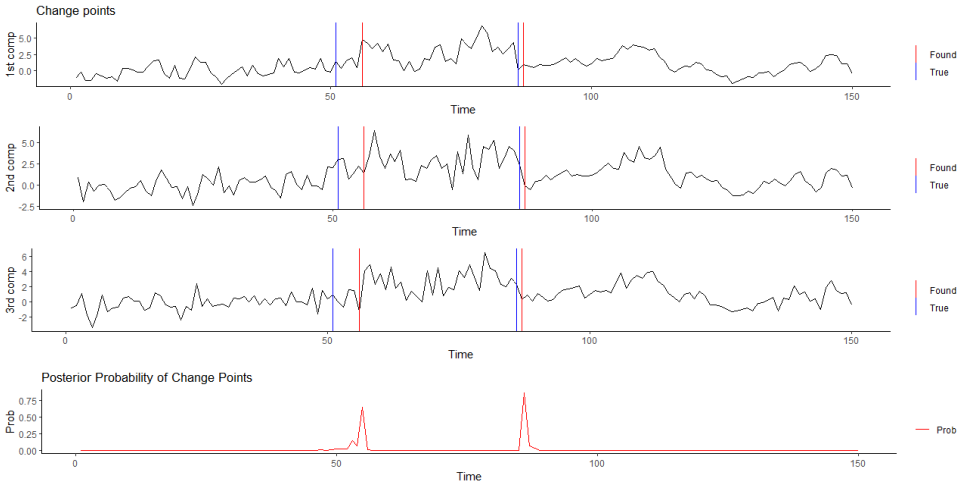

Figure 5: Realization of the simulated process, with the 3 components represented separately. In blue we highlighted the true change points and in red the one we produced with our algorithm. The fourth graph is the posterior probability for each point to be a change point.

We propose a similar plot as the one shown before. This time the performance is not perfect: the mode of simulated change points differs from the real ones, but we still get very close, and, as importantly, we still predict the correct number of change points. This result is still very satisfying because the particular realization that we obtained when simulating the Ornstein-Uhlenbeck process seems extremely challenging for the problem of change point detection, and even a human eye could be easily fooled.

The posterior probabilities, in this case, don't help much: we have spikes but there is not much spread, so this means that the algorithm is confident in the points that proposes. We initially worried that this could have been a problem caused by an insufficient mixing in the MCMC algorithm, but it appeared clear after a few checks that it was a non-recurring problem, so we attributed this behaviour to the particular realization of the simulated process.

4.4 Simulated data conclusions

Thanks to our test on simulated data, we have proved some characteristics of our algorithm that can be summarised as follows:

- The algorithm is very good at finding change points in simulated time series.
- The algorithm is robust with respect to parameters and initial values.
- The algorithm is robust with respect to different type of change points.

5 Real data applications

Now our model is ready to be tested on real world applications, it is suited for several situations, but we want to investigate 2 particular fields: epidemiology and finance. In these fields time series are used often and finding change points is very useful for different goals. The stationarity assumption is broken here, nonetheless our propose performs well even in these cases as will see below. We are really satisfied with this result and we think is one of the strengths of the model.

5.1 Covid-19 cases in north, center and south Italy

We collected data as new daily Covid-19 cases in Italy in the period ranging from 24 *February* 2020 to 29 *November* 2020.

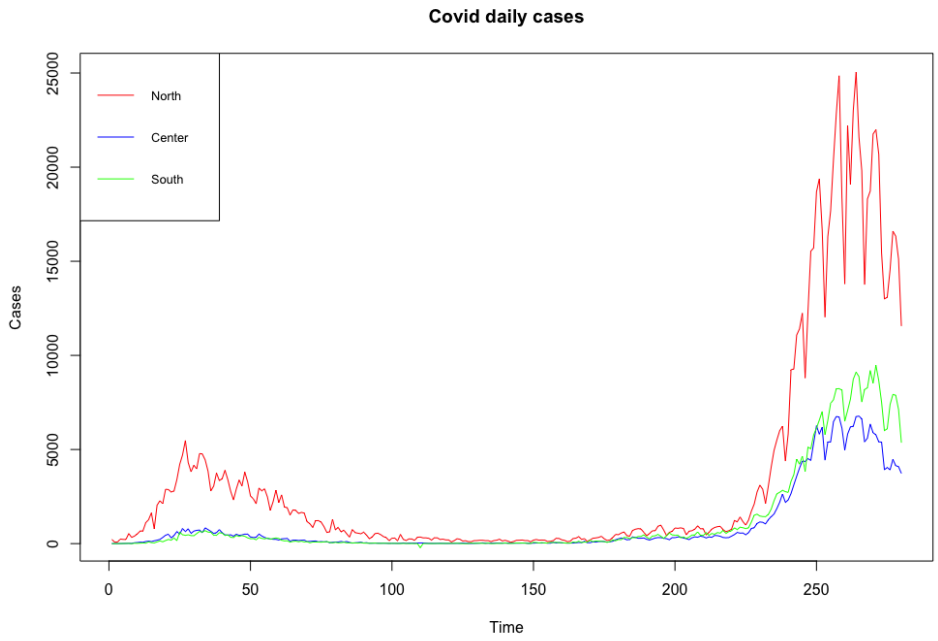

Figure 6: Covid-19 daily cases in the three different parts of Italy, during the period from 24 February 2020 to 29 November 2020

Data is divided into three macroregions (north, center, south), so it is well suited for our proposal. In this case the time series follow similar behaviours over time, however the multivariate aspect is nonetheless useful since it's more informative. Parameters tuning is relevant to achieve perfect results, however a major positive aspect of our proposal is the robustness with respect to parameters and initialization. Thanks to the hierarchical structure of the model, it is very flexible, so a great part of the work is done by the model itself.

We used, for the algorithm, the following parameters initialization:
$a = 1, b = 1, c = 1, d = 1, \theta_0 = 0.1897, \sigma_0 = 0.1, m_0 = mean(\underline{y}), \nu_0 = 4, k_0 = 1, \Psi_0 = var(\underline{y}), \gamma_0 = 0.5, \underline{n}^{initial} = (70, 70, 70, 70), k^{initial} = 4, q = 0.5, N_{rep} = 20000, N_{burnin} = 5000$

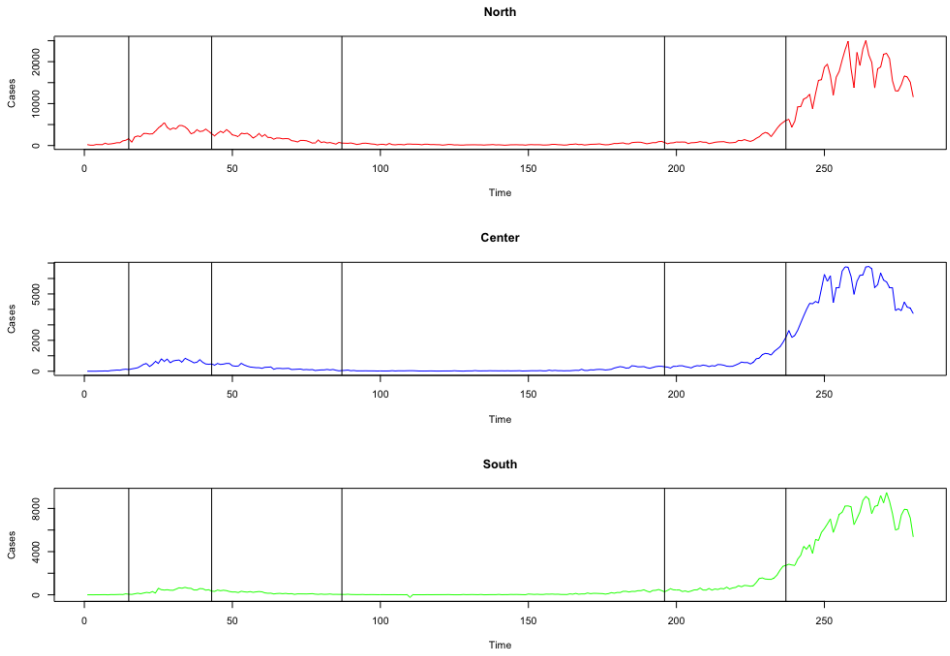

Figure 7: Covid cases in Italy with highlighted the change points we estimated from our MCMC sample using the mode statistic.

Figure 7 shows the detected change points by the mode estimate. The algorithm is able to identify

5 change points, all "correctly" positioned in the plot. In fact where we see the black vertical lines indicating change points there are ascents or descents, and the locations are similar to the one a human would choose.

There's a very accurate correspondence of the estimated points in figure 7 with important events happened or measures taken during the pandemic. This validates our model in real world settings.

- 09 *March* 2020 $\Rightarrow$ First outbreak of the disease. It was the day we realized the severity of the situation and the first measures were taken. We see the first spike in the plot
- 06 *April* 2020 $\Rightarrow$ Two weeks after the first lockdown. Two weeks are needed to see substantial results of DPCMs. We see the start of the gradual drop in the plot
- 20 *May* 2020 $\Rightarrow$ Two weeks from phase-2 DPCM. Infection stabilizes to a low level from here to the summer
- 06 *September* 2020 $\Rightarrow$ Second start of the outbreak of the disease. Summer holidays spread again the virus through Italy and we see a slow increase
- 17 *October* 2020 $\Rightarrow$ Moment of maximum growth of cases. Due to the contagion closure of activities was ordered

As we anticipated before the mode is a naive estimate for the best partition, so we investigate further methods to perform better statistics. Thanks to our computations of posterior probabilities for each time step, we propose another method: set a threshold and consider as a change point every time step that has a posterior probability over it. Then, if we find more than one time step with a relevant probability in a specific neighborhood, we retain only the one with the highest probability.

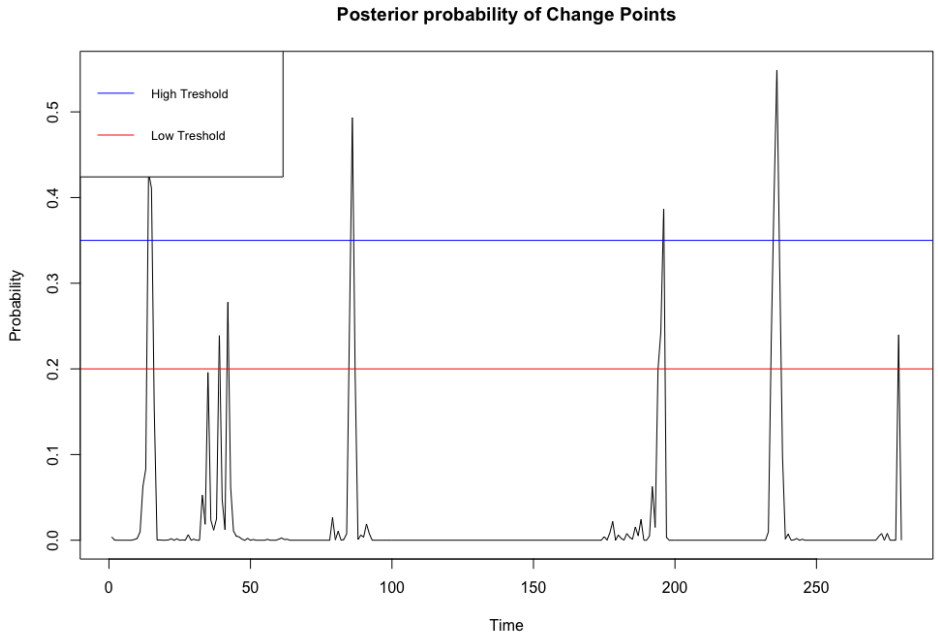

Figure 8: Posterior probability for each point to be a change point for Covid-19 cases in Italy. Two possible thresholds are highlighted

We have two major benefits with this method. The first one is the higher robustness of the estimate, that is very evident in the finance application. The second and more important one is the possibility to choose a posteriori the number of change points and vary it in a hierarchical manner. See Figure 8, we can start with a threshold 1 and lower it progressively to add change points. We cover first the point at $t = 237$, that's obviously the most important one, and then add the others one after the other. It's the same concept of hierarchical clustering with the point where we cut the dendrogram, here the threshold is the cutting point. The "high" and "low" threshold in Figure 8 are 2 possible choices. This greatly improves flexibility, interpretability and can be used to order the change points in a hierarchy of relevance.

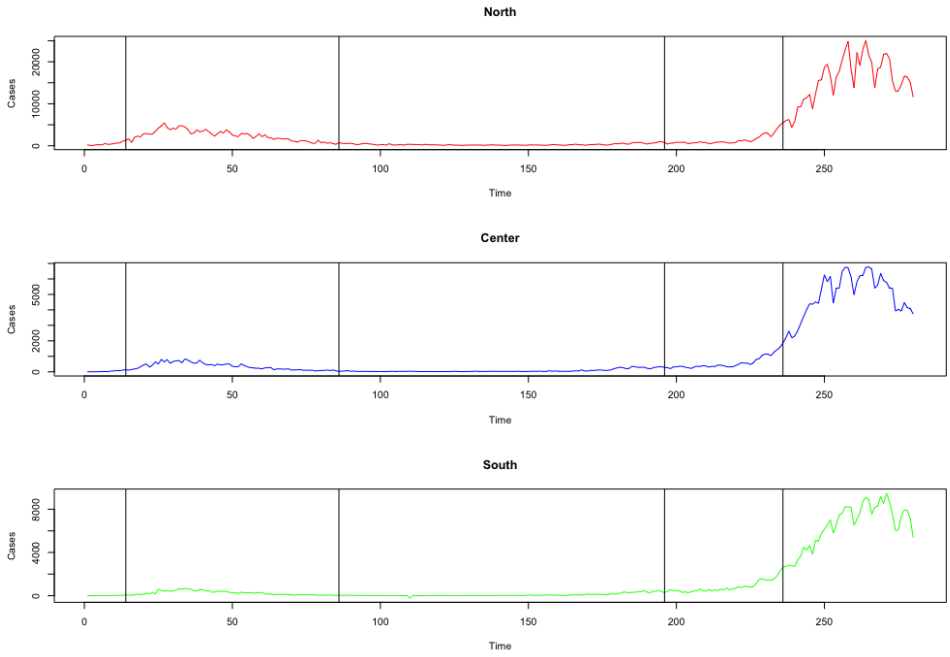

Figure 9: Covid-19 cases with highlighted the change points found using the High threshold

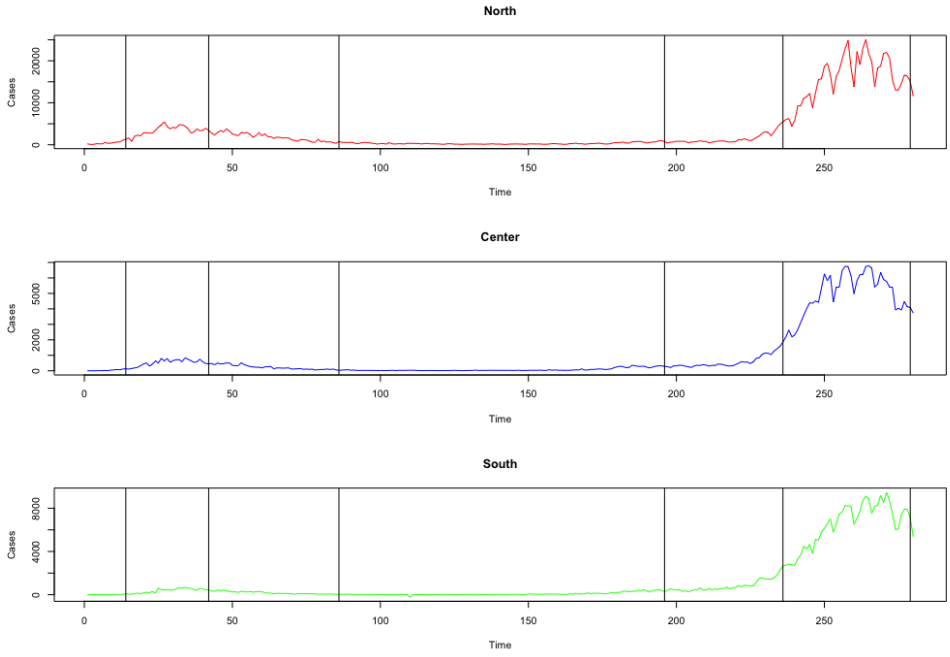

Figure 10: Covid-19 cases with highlighted the change points found using the Low threshold

With the "high" threshold we don't consider as a change point the one at the start of the descent of the first peak (Figure 9) while with the "low" threshold we find also the start of the decrease of the second wave (Figure 10).

5.2 Closing prices of Milan stock exchange companies

Here we chose as dataset the 5 greatest capitalization companies in the Milan stock exchange to describe the behaviour of the market. There are daily closing price from 1 $January$ 2020 to 29 $November$ 2020 for ENEL, Intesa Sanpaolo, Ferrari, ENI, STMicroelectronics. Moreover we collected FTSE MIB index data in the same days to be able to show results with a plot condensing information from all of them. FTSE

MIB index data is not directly used in the multivariate model, it will be used for comparison with the univariate case later.

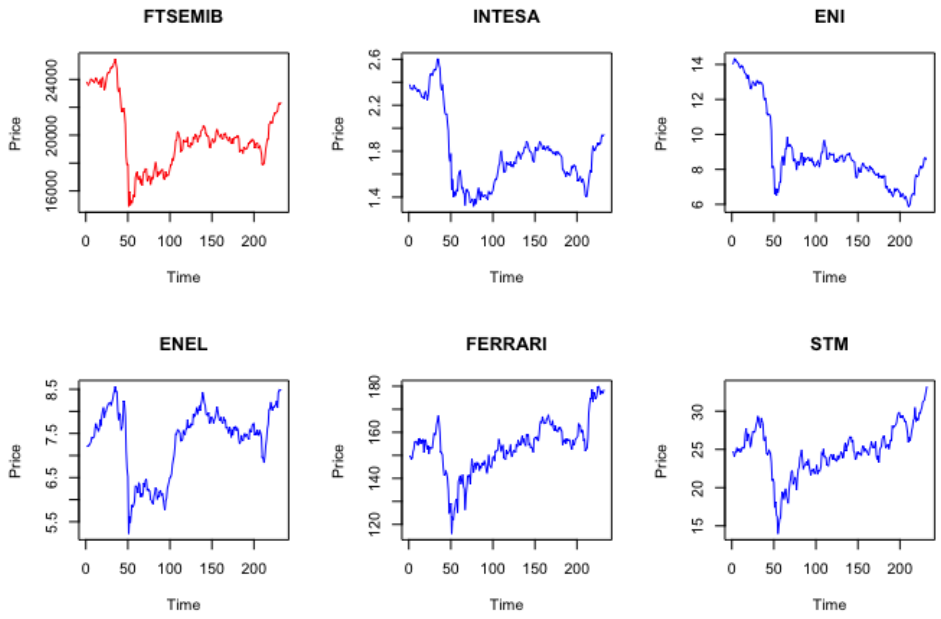

Figure 11: Daily closing prices for the 5 greatest capitalization companies in the Milan Stock Exchange, from 1 January 2020 to 29 November 2020

In Figure 11 we show the time series. This data is much more challenging than the Covid-19 one since we can clearly see it's way less predictable and wigglier. Moreover here the fact that we have available multivariate data is way more useful since some behaviours are shared by all the series, but others that we will detect are specific of certain companies.

We used, for the algorithm, the following parameters initialization:
$a = 1, b = 1, c = 1, d = 1, \theta_0 = 0.1897, \sigma_0 = 0.1, m_0 = mean(\underline{y}), \nu_0 = 5, k_0 = 10, \Psi_0 = var(\underline{y}), \gamma_0 = 0.5, \underline{n}^{initial} = (58, 58, 58, 58), k^{initial} = 4, q = 0.5, N_{rep} = 20000, N_{burnin} = 5000$

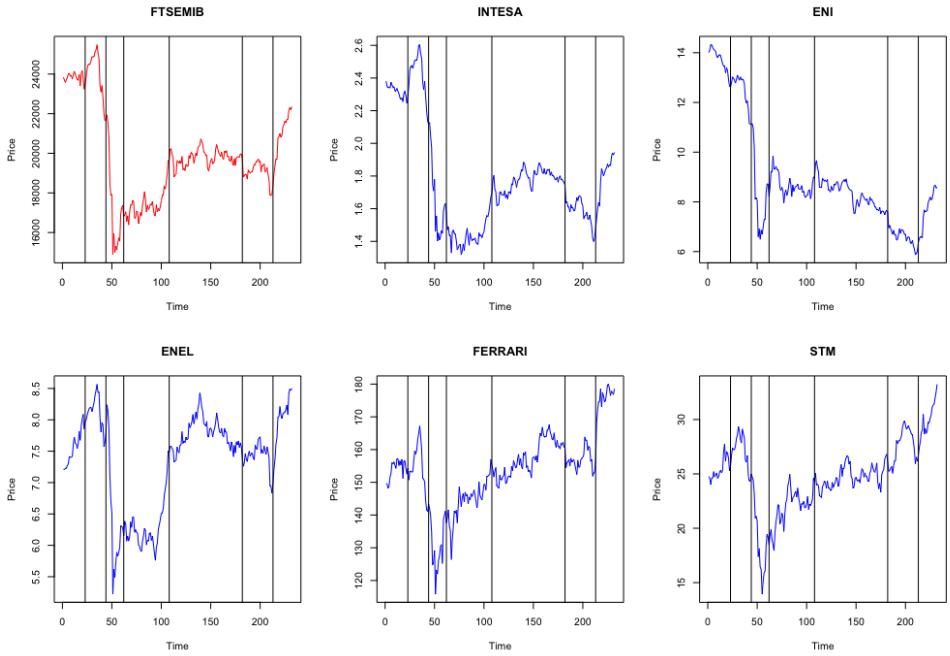

Figure 12: Change points obtained with our threshold technique

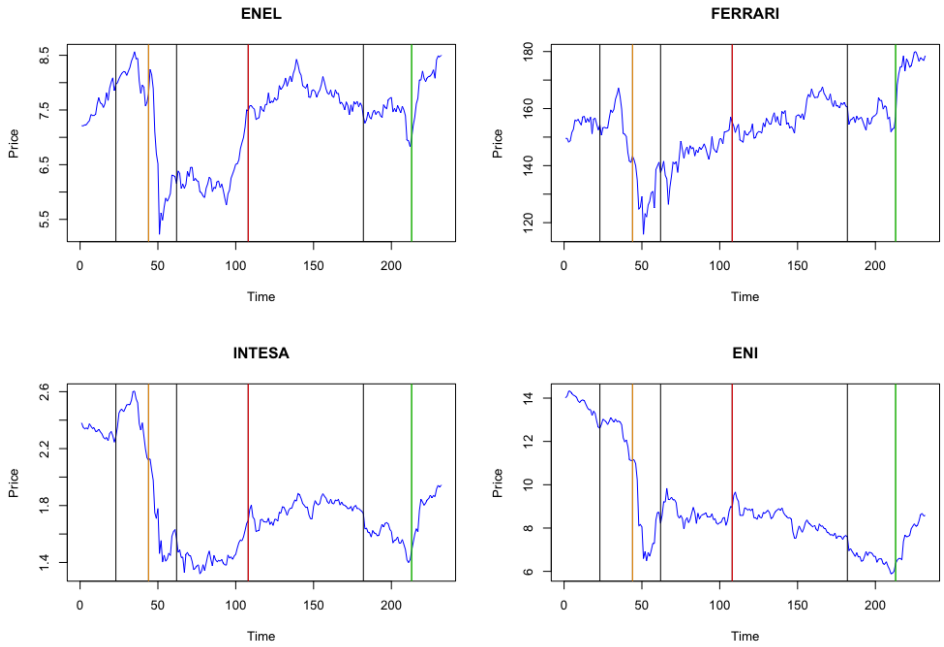

Figure 13: Selection of the plots shown before, with some of the change points highlighted in color for easier visualization

In Figure 12 we show all the time series with the found change points. They are placed in all of the sensible places.

In Figure 13 we focus on some of those for visualization purposes to examine interesting facts. In particular we want to highlight 3 change points. The green one is a time step in which a change occurs for every series, it is easy to detect precisely. The red one is a time step in which a change occurs only for some series, we have it very evident in ENEL, clearly visible in Intesa, but absent in Ferrari and ENI. According to our purposes we easily detect it, it may would have been difficult to find it with a univariate approach depending on the specific series chosen. The orange one is clear to be a change point from any plot, since it represents the start of the steepest part of the descent, however its exact location is difficult to grasp. Nevertheless leveraging on the fact that having multivariate data we have more information, we can borrow strength from the ENEL data, in which the point is more marked, and we are able to precisely find it. In each one of the very different previous cases, the algorithm is able to precisely detect the relevant change points.

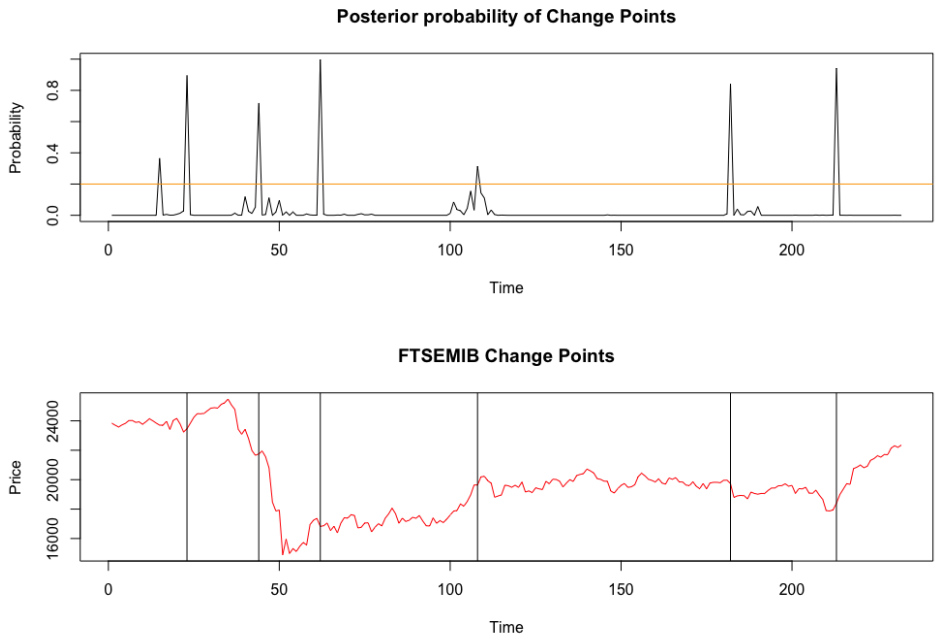

Figure 14: FTSE MIB index series with the posterior probabilities corresponding to the change points highlighted.

With this problem is essential to use the more robust estimate to find the change points, otherwise we would be more prone to substantial errors. In Figure 14 there are the posterior probabilities and the optimal change points. Also in this case is possible to lower or rise the threshold according to our goals.

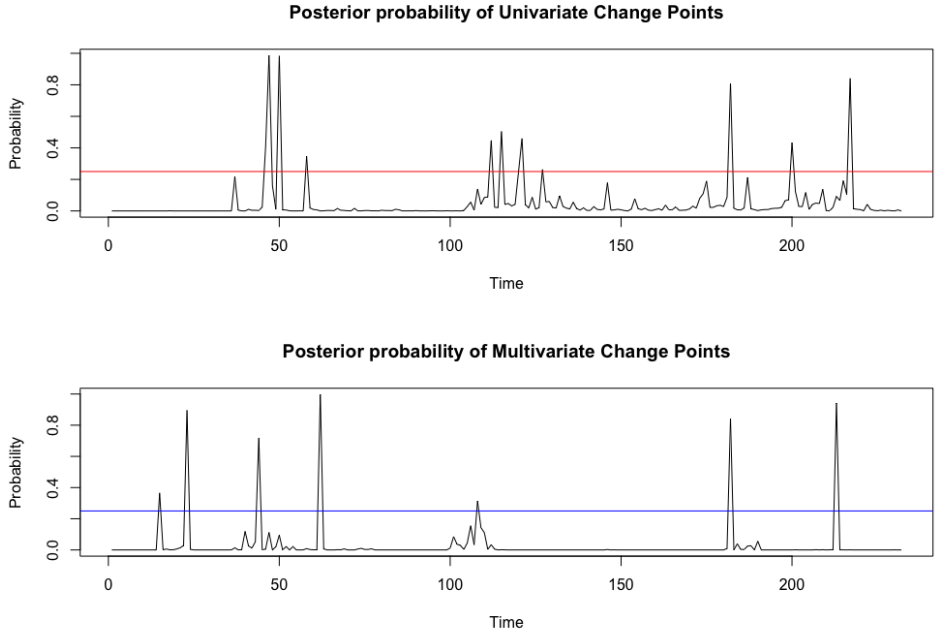

Figure 15: Posterior probabilities of the change points when using univariate data against multivariate data.

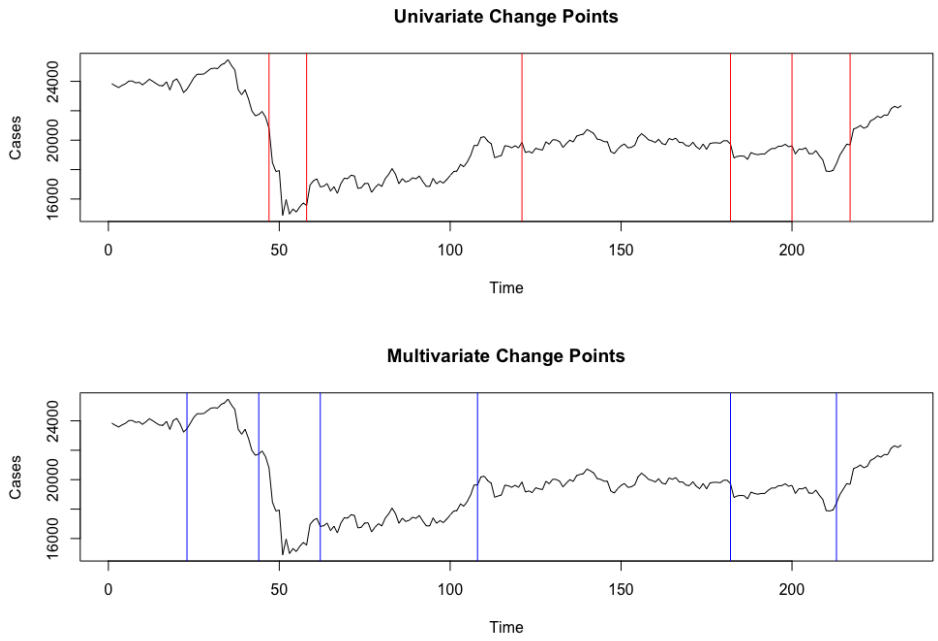

Figure 16: Change points detected using univariate data against multivariate data.

To conclude this application we compare our results to the ones from the univariate approach employed on the FTSE MIB index, considered as the best way to take into account for these companies together. First fact noticed from Figure 15 is that the univariate algorithm is much more uncertain about the position of change points, in fact there are more "neighbor" spikes that probably account for the same point but are classified as different. Moreover in Figure 16 we see that the univariate change point at $t = 200$ is probably not present while the one at $t = 121$ is shifted to the right. Both of these problems are absent with the multivariate approach that didn't even use the FTSE MIB data directly.

5.3 Real data conclusions

Thanks to real world applications we can properly highlight the advantages and disadvantages of our proposal:

- The model shows a high robustness with respect to parameters choice, so that is easier to tune it. Parameters help to achieve accurate detection, however with standard choices is often possible to achieve good results.
- With the probability threshold estimate we improve interpretability and flexibility a posteriori, resulting in an attractive technique when the importance of change points is relevant, since is possible to easily compute it.
- The correspondence with important events further validates the soundness of the model, because it achieves the main goal of detecting when underlying events influencing data happened.
- Particularly when components are very different, the higher quantity of information results in a sensible improvement of performances with respect to the univariate case.
- There are also some drawbacks with a multivariate extension, in particular the higher computational time requested, and the lower maximum size possible. We think that usually our proposal, although it is applicable in less situations, when feasible, achieves better results.

6 Discussion

In this report we proposed a change point detection model which extends to the multivariate case the work of Martinez and Mena (2014). We started from their EPPF model preserving time ordering for the prior that results in a Pitman-Yor process and we added hyperpriors for further robustness. We used their idea of integrated regime likelihood for the Ornstein-Uhlenbeck process and extended it by computing it in the multivariate case. To simulate from the posterior we leveraged their split-merge MCMC algorithm that we improved and sped up by simplifying ratios of posterior probabilities. We validated the model with simulated data to show performances in several situations and highlighted its strengths. Finally we tested the approach on real data and it gave explainable and accurate results in two very different applications of great interest.

The change point detection problem is key in most of the anomaly detection applications, so it is relevant and fast growing. We think our model is well suited in several contexts of this problem in which precision and interpretability are essential. For future work some ideas of improvement are to extend this approach to cope with more data, hoping to maintain enough accuracy in the process, or to speed the MCMC, thanks to a better implementation.

References

Fuentes–García, R., Mena, R. H., & Walker, S. G. (2010). A probability for classification based on the dirichlet process mixture model. *Journal of Classification*, *27*(3), 389—403. DOI: 10.1007/s00357-010-9061-9

Martinez, F., & Mena, R. (2014, 04). On a nonparametric change point detection model in markovian regimes. *Bayesian Analysis*, *TBA*. DOI: 10.1214/14-BA878

Pitman, J. (2006). *Combinatorial stochastic processes.* Ecole d'eté de probabilités de Saint-Flour XXXII - 2002. Springer.

Uhlenbeck, G. E., & Ornstein, L. S. (1930, 09). On the theory of the brownian motion. *Phys. Rev.*, *36*, 823–841. DOI: 10.1103/PhysRev.36.823

Vatiwutipong, P., & Phewchean, N. (2019, 07). Alternative way to derive the distribution of the multivariate ornstein–uhlenbeck process. *Advances in Difference Equations*, *2019*. DOI: 10.1186/s13662-019-2214-1

Supplementary Files

All the code used for the studies in this paper can be found at the git repository `https://github.com/piergiuseppepezzoli/On-a-bayesian-change-point-detection-model-for-multivariate-data.git`