The DRPM Strikes Back: More Flexibility for a Bayesian Spatio-Temporal Clustering Model

Candidate: Federico Angelo Mor

Advisor: Prof. Alessandra Guglielmi
Coadvisor: Prof. Alessandro Carminati

POLITECNICO
MILANO 1863

December 11, 2024

What is the thesis about?

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the dependencies in the sequence of clusters over time.

Currently, the model

- produces up to spatially-informed clusters
- only accepts complete datasets
- has quite slow execution times (especially on large datasets)

What is the thesis about?

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the dependencies in the sequence of clusters over time.

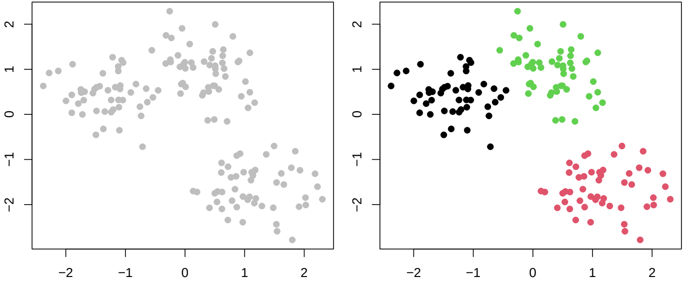
Currently, the model

- produces up to spatially-informed clusters
  → I additionally introduced covariates information
- only accepts complete datasets
  → I made it work with missing values in the target variable
- has quite slow execution times (especially on large datasets)
  → I developed a brand-new and more efficient implementation

Clustering

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

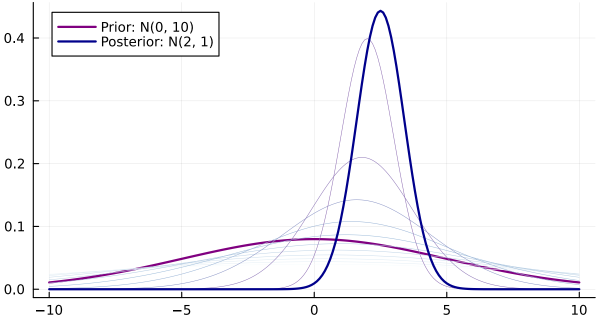
Clustering is a fundamental technique of unsupervised learning where a set of data points has to be divided into homogeneous groups of units which exhibit a similar behaviour.



Why going Bayesian?

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

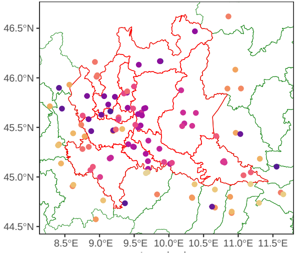
Bayesian models incorporate prior information on the model parameters and allow to assess uncertainty when performing inference on the results.

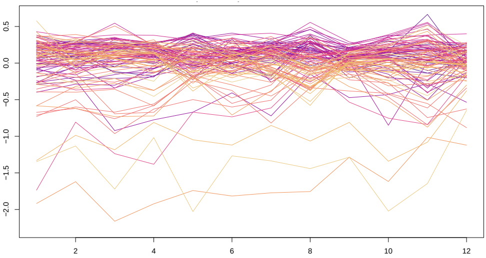


A bit of (spatio-temporal) context

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

In spatio-temporal datasets, observations are collected over time and across various spatial locations. So we will have n units that have to be clustered at all time instants $t = 1, \ldots, T$.

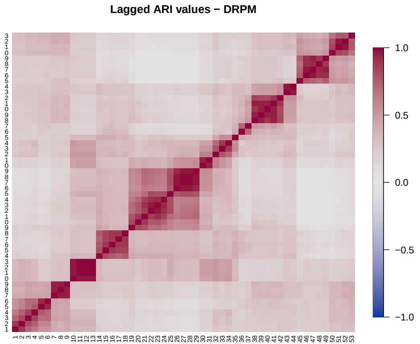


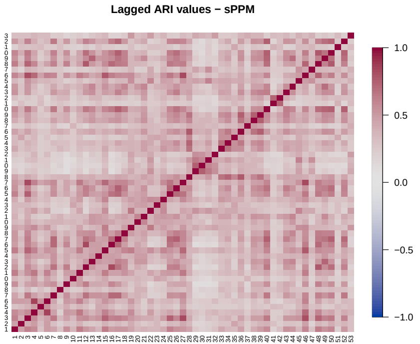


Why should we care about temporal dependencies?

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

This allows to derive a more gentle and interpretable evolution of clusters.

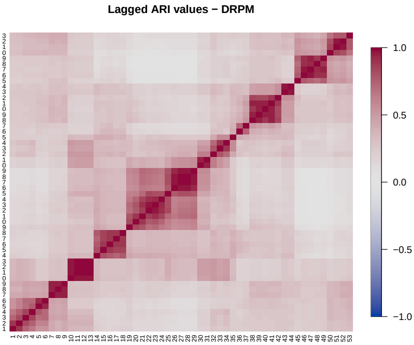


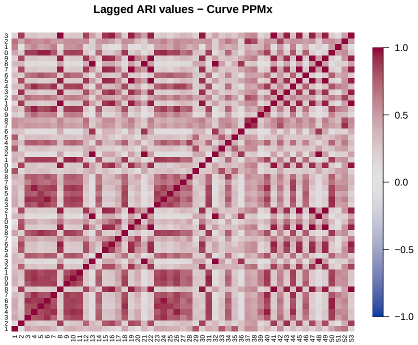


Why should we care about temporal dependencies?

The Dependent Random Partition Model from (**1-drpm**) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

This allows to derive a more gentle and interpretable evolution of clusters.





Modelling the temporal dependence

Introducing temporal dependence in a collection of partitions requires the formulation of a joint probability model for $(\rho_1, \ldots, \rho_T)$. **1-drpm** modelled this temporal connection as a first-order Markovian structure, where the conditional distribution of ρ_t given all the predecessors $\rho_{t-1}, \rho_{t-2}, \ldots, \rho_1$ actually depends only on ρ_{t-1}, leading to

$$P(\rho_1, \ldots, \rho_T) = P(\rho_T|\rho_{T-1}) \cdots P(\rho_2|\rho_1)P(\rho_1) \quad (1)$$

Here, $P(\rho_1)$ is an exchangeable partition probability function (EPPF), which describes how the n experimental units at time period 1 are grouped into k_1 distinct clusters. **1-drpm** chose this EPPF to be $P(\rho_1) \propto \prod_{j=1}^{k_1} M \cdot (|S_{j1}| - 1)!$.

Modelling the temporal dependence

To characterize the other terms $P(\rho_t|\rho_{t-1})$ in (1), i.e. to explicitly model how ρ_{t-1} influences ρ_t, the following auxiliary variables need to be introduced to. For all units $i = 1, \ldots, n$ we define

$$\gamma_{it} = \begin{cases} 1 & \text{if unit } i \text{ is } \textit{not} \text{ reallocated when moving from time } t-1 \text{ to } t \\ 0 & \text{otherwise (namely, the unit } \textit{is} \text{ reallocated)} \end{cases}$$

These parameters model the similarity between ρ_{t-1} and ρ_t:

- if ρ_{t-1} and ρ_t are highly dependent, their cluster configurations will change minimally $\implies$ the majority of γ_{it} will be 1
- if ρ_{t-1} and ρ_t exhibit low dependence, their cluster configurations will change significantly $\implies$ the majority of γ_{it} will be 0

Modelling the temporal dependence

1-drpm assumed $\gamma_{it} \overset{\text{ind}}{\sim} \text{Ber}(\alpha_t)$, where $\alpha_t \in [0,1]$ serves as a temporal dependence parameter, spanning from perfect temporal correlation ($\alpha_t = 0$) to full independence ($\alpha_t = 1$).

For clarity, the vector $\boldsymbol{\gamma}_t = (\gamma_{1t}, \ldots, \gamma_{nt})$ is introduced, so that the T pairs of parameters $(\boldsymbol{\gamma}_j, \rho_j)$ are explicitly reported in the augmented formulation of the joint model (1), which becomes

$$P(\boldsymbol{\gamma}_1, \rho_1, \ldots, \boldsymbol{\gamma}_T, \rho_T) = P(\rho_T | \boldsymbol{\gamma}_T, \rho_{T-1}) P(\boldsymbol{\gamma}_T) \cdots \\ P(\rho_2 | \boldsymbol{\gamma}_2, \rho_1) P(\boldsymbol{\gamma}_2) P(\rho_1) \tag{2}$$

Once the model for the partition is specified, there is considerable flexibility in how to define the remainder of the Bayesian model.

The DRPM

DRPM formulation according to **1-drpm** (henceforth, CDRPM, with C as the C language used for the model's implementation).

$$
\begin{aligned}
Y_{it} \mid Y_{it-1}, \boldsymbol{\mu}_t^\star, \boldsymbol{\sigma}_t^{2\star}, \boldsymbol{\eta}, \boldsymbol{c}_t &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{it}t} + \eta_{1i} Y_{it-1}, \sigma^{2\star}_{c_{it}t}(1-\eta_{1i}^2)) \\
Y_{i1} &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{i1}1}, \sigma^{2\star}_{c_{i1}1}) \\
\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i}+1)) &\overset{\text{ind}}{\sim} \text{Laplace}(a, b) \\
(\mu^\star_{jt}, \sigma^\star_{jt}) &\overset{\text{ind}}{\sim} \mathcal{N}(\vartheta_t, \tau_t^2) \times \mathcal{U}(0, A_\sigma) \\
\vartheta_t \mid \vartheta_{t-1} &\overset{\text{ind}}{\sim} \mathcal{N}((1-\varphi_1)\varphi_0 + \varphi_1 \vartheta_{t-1}, \lambda^2(1-\varphi_1^2)) \\
(\vartheta_1, \tau_t) &\overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \mathcal{U}(0, A_\tau) \\
(\varphi_0, \varphi_1, \lambda) &\sim \mathcal{N}(m_0, s_0^2) \times \mathcal{U}(-1, 1) \times \mathcal{U}(0, A_\lambda) \\
\{\boldsymbol{c}_t, \ldots, \boldsymbol{c}_T\} &\sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)
\end{aligned}
$$

The DRPM - the autoregressive parameters

To facilitate the propagation of temporal dependence throughout the model, an autoregressive AR(1) component is incorporated at both the data level and the cluster-specific parameter level.

$$
\begin{aligned}
Y_{it} | Y_{it-1}, \boldsymbol{\mu}_t^{\star}, \boldsymbol{\sigma}_t^{2\star}, \boldsymbol{\eta}, \boldsymbol{c}_t &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^{\star}_{c_{it}t} + \boxed{\eta_{1i} Y_{it-1}}, \sigma^{2\star}_{c_{it}t}(1-\eta_{1i}^2)) \\
Y_{i1} &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^{\star}_{c_{i1}1}, \sigma^{2\star}_{c_{i1}1}) \\
\boxed{\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i}+1))} &\overset{\text{ind}}{\sim} \text{Laplace}(a, b) \\
(\mu^{\star}_{jt}, \sigma^{\star}_{jt}) &\overset{\text{ind}}{\sim} \mathcal{N}(\vartheta_t, \tau_t^2) \times \mathcal{U}(0, A_\sigma) \\
\vartheta_t | \vartheta_{t-1} &\overset{\text{ind}}{\sim} \mathcal{N}((1-\varphi_1)\varphi_0 + \boxed{\varphi_1 \vartheta_{t-1}}, \lambda^2(1-\varphi_1^2)) \\
(\vartheta_1, \tau_t) &\overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \mathcal{U}(0, A_\tau) \\
(\varphi_0, \boxed{\varphi_1}, \lambda) &\sim \mathcal{N}(m_0, s_0^2) \times \mathcal{U}(-1, 1) \times \mathcal{U}(0, A_\lambda) \\
\{\boldsymbol{c}_t, \ldots, \boldsymbol{c}_T\} &\sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)
\end{aligned}
$$

The DRPM - the ϑ_t parameter

The ϑ_t parameter serves as a temporal anchor for the cluster-specific means $\mu^\star_{jt}$, ensuring that these means are not completely independent over time but instead exhibit a regular and interpretable progression.

$$
\begin{aligned}
Y_{it} | Y_{it-1}, \boldsymbol{\mu}^\star_t, \boldsymbol{\sigma}^{2\star}_t, \boldsymbol{\eta}, \boldsymbol{c}_t &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{it}t} + \eta_{1i} Y_{it-1}, \sigma^{2\star}_{c_{it}t}(1 - \eta^2_{1i})) \\
Y_{i1} &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{i1}1}, \sigma^{2\star}_{c_{i1}1}) \\
\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i} + 1)) &\overset{\text{ind}}{\sim} \text{Laplace}(a, b) \\
(\mu^\star_{jt}, \sigma^\star_{jt}) &\overset{\text{ind}}{\sim} \mathcal{N}(\boxed{\vartheta_t}, \tau^2_t) \times \mathcal{U}(0, A_\sigma) \\
\boxed{\vartheta_t | \vartheta_{t-1}} &\overset{\text{ind}}{\sim} \mathcal{N}((1 - \varphi_1)\varphi_0 + \varphi_1 \boxed{\vartheta_{t-1}}, \lambda^2(1 - \varphi^2_1)) \\
(\boxed{\vartheta_1}, \tau_t) &\overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \mathcal{U}(0, A_\tau) \\
(\varphi_0, \varphi_1, \lambda) &\sim \mathcal{N}(m_0, s^2_0) \times \mathcal{U}(-1, 1) \times \mathcal{U}(0, A_\lambda) \\
\{\boldsymbol{c}_t, \ldots, \boldsymbol{c}_T\} &\sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)
\end{aligned}
$$

Our generalized model

DRPM formulation according to our generalization (henceforth, JDRPM, with J as the Julia language used for the model's implementation).

$$
\begin{aligned}
Y_{it} | Y_{it-1}, \boldsymbol{\mu}_t^\star, \boldsymbol{\sigma}_t^{2\star}, \boldsymbol{\eta}, \boldsymbol{c}_t &\overset{\text{ind}}{\sim} \mathcal{N}(\mu_{c_{it}t}^\star + \eta_{1i} Y_{it-1} + \boldsymbol{x}_{it}^T \boldsymbol{\beta}_t, \sigma_{c_{it}t}^{2\star}(1 - \eta_{1i}^2)) \\
Y_{i1} &\overset{\text{ind}}{\sim} \mathcal{N}(\mu_{c_{i1}1}^\star + \boldsymbol{x}_{i1}^T \boldsymbol{\beta}_1, \sigma_{c_{i1}1}^{2\star}) \\
\boldsymbol{\beta}_t &\overset{\text{ind}}{\sim} \mathcal{N}_p(\boldsymbol{b}, s^2 I) \\
\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i} + 1)) &\overset{\text{ind}}{\sim} \text{Laplace}(a, b) \\
(\mu_{jt}^\star, \sigma_{jt}^{2\star}) &\overset{\text{ind}}{\sim} \mathcal{N}(\vartheta_t, \tau_t^2) \times \text{invGamma}(a_\sigma, b_\sigma) \\
\vartheta_t | \vartheta_{t-1} &\overset{\text{ind}}{\sim} \mathcal{N}((1 - \varphi_1)\varphi_0 + \varphi_1 \vartheta_{t-1}, \lambda^2(1 - \varphi_1^2)) \\
(\vartheta_1, \tau_t^2) &\overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \text{invGamma}(a_\tau, b_\tau) \\
(\varphi_0, \varphi_1, \lambda^2) &\sim \mathcal{N}(m_0, s_0^2) \times \mathcal{U}(-1, 1) \times \text{invGamma}(a_\lambda, b_\lambda) \\
\{\boldsymbol{c}_t, \dots, \boldsymbol{c}_T\} &\sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)
\end{aligned}
$$

Updated formulation - regression in the likelihood

(1) We inserted a regression term in the likelihood and changed the prior distributions of the variance parameters

$$
\begin{aligned}
Y_{it} | Y_{it-1}, \boldsymbol{\mu}_t^\star, \boldsymbol{\sigma}_t^{2\star}, \boldsymbol{\eta}, \boldsymbol{c}_t &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{it}t} + \eta_{1i} Y_{it-1} + \boldsymbol{x}_{it}^T \beta_t, \sigma^{2\star}_{c_{it}t}(1-\eta_{1i}^2)) \\
Y_{i1} &\overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{i1}1} + \boldsymbol{x}_{i1}^T \beta_1, \sigma^{2\star}_{c_{i1}1}) \\
\beta_t &\overset{\text{ind}}{\sim} \mathcal{N}_p(\boldsymbol{b}, s^2 I) \\
\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i}+1)) &\overset{\text{ind}}{\sim} \text{Laplace}(a, b) \\
(\mu^\star_{jt}, \sigma^{2\star}_{jt}) &\overset{\text{ind}}{\sim} \mathcal{N}(\vartheta_t, \tau_t^2) \times \text{invGamma}(a_\sigma, b_\sigma) \\
\vartheta_t | \vartheta_{t-1} &\overset{\text{ind}}{\sim} \mathcal{N}((1-\varphi_1)\varphi_0 + \varphi_1 \vartheta_{t-1}, \lambda^2(1-\varphi_1^2)) \\
(\vartheta_1, \tau_t^2) &\overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \text{invGamma}(a_\tau, b_\tau) \\
(\varphi_0, \varphi_1, \lambda^2) &\sim \mathcal{N}(m_0, s_0^2) \times \mathcal{U}(-1, 1) \times \text{invGamma}(a_\lambda, b_\lambda) \\
\{\boldsymbol{c}_t, \ldots, \boldsymbol{c}_T\} &\sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)
\end{aligned}
$$

Updated formulation

The regressor term $\boldsymbol{\beta}_t$ improves the quality of the fitted estimates for the target variable and for the samples of the model parameters.

Prior: $\boldsymbol{\beta}_t \sim \mathcal{N}_p(\boldsymbol{b}, s^2 I)$
Update rule:

for $t = 1$: $f(\boldsymbol{\beta}_t|-) \propto$ kernel of a $\mathcal{N}\text{Canon}(\boldsymbol{h}_{(\text{post})}, J_{(\text{post})})$ with

$$\boldsymbol{h}_{(\text{post})} = \left(\frac{\boldsymbol{b}}{s^2} + \sum_{i=1}^{n} \frac{(Y_{it} - \mu^{\star}_{c_{it}t})\boldsymbol{x}_{it}}{\sigma^{2\star}_{c_{it}t}} \right) \quad J_{(\text{post})} = \left(\frac{1}{s^2} I + \sum_{i=1}^{n} \frac{\boldsymbol{x}_{it}\boldsymbol{x}_{it}^T}{\sigma^{2\star}_{c_{it}t}} \right)$$

for $t > 1$: $f(\boldsymbol{\beta}_t|-) \propto$ kernel of a $\mathcal{N}\text{Canon}(\boldsymbol{h}_{(\text{post})}, J_{(\text{post})})$ with

$$\boldsymbol{h}_{(\text{post})} = \left(\frac{\boldsymbol{b}}{s^2} + \sum_{i=1}^{n} \frac{(Y_{it} - \mu^{\star}_{c_{it}t} - \eta_{1i} Y_{it-1})\boldsymbol{x}_{it}}{\sigma^{2\star}_{c_{it}t}} \right) \quad J_{(\text{post})} = \left(\frac{1}{s^2} I + \sum_{i=1}^{n} \frac{\boldsymbol{x}_{it}\boldsymbol{x}_{it}^T}{\sigma^{2\star}_{c_{it}t}} \right)$$

where $\mathcal{N}\text{Canon}(\boldsymbol{h}, J)$ is the canonical formulation of the $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$, with $\boldsymbol{h} = \Sigma^{-1}\boldsymbol{\mu}$ and $J = \Sigma^{-1}$.

Updated formulation - the variances' distribution

The choice of the inverse gamma distribution recovers conjugacy within the model, leading to better mixing properties for the MCMC.

Prior: $\sigma^{2\star}_{jt} \sim \text{invGamma}(a_\sigma, b_\sigma)$
Update rule:

for $t = 1$: $f(\sigma^{2\star}_{jt} | -) \propto$ kernel of a $\text{invGamma}(a_{\sigma(\text{post})}, b_{\sigma(\text{post})})$ with

$$a_{\tau(\text{post})} = a_\sigma + \frac{|S_{jt}|}{2} \quad b_{\tau(\text{post})} = b_\sigma + \frac{1}{2} \sum_{i \in S_{jt}} (Y_{it} - \mu^\star_{jt} - \boldsymbol{x}^T_{it} \boldsymbol{\beta}_t)^2$$

for $t > 1$: $f(\sigma^{2\star}_{jt} | -) \propto$ kernel of a $\text{invGamma}(a_{\sigma(\text{post})}, b_{\sigma(\text{post})})$ with

$$a_{\tau(\text{post})} = a_\sigma + \frac{|S_{jt}|}{2} \quad b_{\tau(\text{post})} = b_\sigma + \frac{1}{2} \sum_{i \in S_{jt}} (Y_{it} - \mu^\star_{jt} - \eta_{1i} Y_{it-1} - \boldsymbol{x}^T_{it} \boldsymbol{\beta}_t)^2$$

Similar derivations apply to τ^2_t and λ^2.

Additional information level

(2) We introduced covariates information inside the prior for the partition.
To describe how we performed this inclusion, we recall how **1-drpm** included spatial information in the prior for the partition.

The original formulation of the EPPF is

$$P(\rho_t|M) \propto \prod_{j=1}^{k_t} c(S_{jt}|M)$$

where $c(S_{jt}|M)$ describes how units inside cluster S_{jt} are likely to be clustered together a priori.

Spatial information

Let $\boldsymbol{s}_i$ denote the spatial coordinates of the i-th unit (noting that these coordinates do not change over time), and $\boldsymbol{s}^\star_{jt}$ denote the subset of spatial coordinates of the units belonging to cluster S_{jt}. Then, we can express the EPPF for the t-th partition in the form

$$P(\rho_t|M,\mathcal{S}) \propto \prod_{j=1}^{k_t} C(S_{jt},\boldsymbol{s}^\star_{jt}|M,\mathcal{S})$$

where the cohesion function $C(S_{jt},\boldsymbol{s}^\star_{jt}|M,\mathcal{S})$, parametrised by a set of parameters $\mathcal{S}$, measures the compactness of the spatial coordinates $\boldsymbol{s}^\star_{jt}$.

Covariates information

Let $X^\star_{jt}$ denote the $p \times |S_{jt}|$ matrix that contains the covariates of the units belonging to cluster S_{jt}, i.e. $X^\star_{jt} = \{\boldsymbol{x}^\star_{it} = (x_{it1}, \ldots, x_{itp})^T : i \in S_{jt}\}$. In the current implementation of JDRPM we chose to treat each covariate individually, leading to an EPPF in the form

$$P(\rho_t | M, \mathcal{S}, \mathcal{C}) \propto \prod_{j=1}^{k_t} C(S_{jt}, \boldsymbol{s}^\star_{jt} | M, \mathcal{S}) \left(\prod_{r=1}^{p} g(S_{jt}, \boldsymbol{x}^\star_{jtr} | \mathcal{C}) \right)$$

where the similarity function $g(S_{jt}, \boldsymbol{x}^\star_{jtr} | \mathcal{C})$, parametrised by a set of parameters $\mathcal{C}$, measures the similarity of the r-th covariate values $\boldsymbol{x}^\star_{jtr}$.

Missing data

(3) We let the model accept missing data in the target variable through the derivation of an update rule for the missing Y_{it}'s.

for $t = 1$: $f(Y_{it}|-) \propto \mathcal{N}(\mu_{Y_{it}(\text{post})}, \sigma^2_{Y_{it}(\text{post})})$ with

$$\sigma^2_{Y_{it}(\text{post})} = 1 \Big/ \left(\frac{1}{\sigma^{2\star}_{c_{it}t}} + \frac{\eta^2_{1i}}{2\sigma^{2\star}_{c_{it+1}t+1}(1-\eta^2_{1i})} \right)$$

$$\mu_{Y_{it}(\text{post})} = \sigma^2_{Y_{it}(\text{post})} \left(\frac{\mu^\star_{c_{it}t} + \boldsymbol{x}_{it}^T \boldsymbol{\beta}_t}{\sigma^{2\star}_{c_{it}t}} + \frac{\eta_{1i}(Y_{it+1} - \mu^\star_{c_{it+1}t+1} - \boldsymbol{x}_{it+1}^T \boldsymbol{\beta}_{t+1})}{\sigma^{2\star}_{c_{it+1}t+1}(1-\eta^2_{1i})} \right)$$

for $1 < t < T$: $f(Y_{it}|-) \propto \mathcal{N}(\mu_{Y_{it}(\text{post})}, \sigma^2_{Y_{it}(\text{post})})$ with

$$\sigma^2_{Y_{it}(\text{post})} = \left(1-\eta^2_{1i}\right) \Big/ \left(\frac{1}{\sigma^{2\star}_{c_{it}t}} + \frac{\eta^2_{1i}}{\sigma^{2\star}_{c_{it+1}t+1}} \right)$$

$$\mu_{Y_{it}(\text{post})} = \sigma^2_{Y_{it}(\text{post})} \left(\frac{\mu^\star_{c_{it}t} + \eta_{1i} Y_{it-1} + \boldsymbol{x}_{it}^T \boldsymbol{\beta}_t}{\sigma^{2\star}_{c_{it}t}(1-\eta^2_{1i})} + \frac{\eta_{1i}(Y_{it+1} - \mu^\star_{c_{it+1}t+1} - \boldsymbol{x}_{it+1}^T \boldsymbol{\beta}_{t+1})}{\sigma^{2\star}_{c_{it+1}t+1}(1-\eta^2_{1i})} \right)$$

for $t = T$: $f(Y_{it}|-)$ is just the likelihood of Y_{it}

New implementation

(4) We developed a brand-new and more efficient implementation for the updated MCMC algorithm which we now describe in the following section.

Implementation and optimizations

The MCMC algorithm to compute the posterior samples of our updated model has been implemented in Julia (**Julia-cite**), which have several benefits compared to the C choice of **1-drpm**.

Why Julia?

- Julia combines the ease and expressiveness of high-level languages (e.g. R, python, matlab) with the efficiency and performance characteristics of low-level languages (e.g. C, C++, Fortran)
- Julia code can be tested interactively, as with interpreted languages...
- ... but performance is guaranteed through (just-in-time) compilation
- Linear algebra computations are optimized through BLAS and LAPACK libraries
- There is a vast collection of optimized and complete scientific packages (e.g. `Statistics`, `Distributions`, `MCMCChains`, etc.)

Spoiler alert

Using Julia, we obtained improved computational performance compared to the original C implementation of (**1-drpm**).

This performance gain was achieved through several optimizations steps which we now briefly highlight.

General optimizations

- preallocation of all modelling and working variables
- ensuring type stability of the code, using the package `Cthulhu`

```
indexes::Vector{Int64} = findall((gamma_iter::Matrix{Bool}[::Core.Const(…,t
+1] .== 1)
Si_comp1 = @view rho_tmp::Vector{Int64}[indexes::Vector{Int64}]
Si_comp2 = @view Si_iter::Matrix{Int64}[indexes::Vector{Int64},(t::Int64+1)
::In…]
rho_comp::Int64 = compatibility(Si_comp1::SubArray{Int64, 1, Vector{Int64},
 …, Si_comp2)

if (rho_comp::Int64 != 1)::Bool
    ph::Vector{Float64}[k::Int64] = log(0)::Core.Const(-Inf) # assignment to
    a new cluster is not compatible
else
    # sample new params for this new cluster
    muh_draw::Float64 = rand(Normal(theta_iter::Vector{Float64}[t::Int64]
    ::Float64, sqrt(tau2_iter:…[t])))
    sig2h_draw::Float64 = rand(InverseGamma(sig2h_priors::Vector{Float64}[1]
    ::Float64,sig2h_priors::Vector…[2]))
```

General optimizations

- avoiding unnecessary allocations through the `view` instruction

```
ph[k] = loglikelihood(Normal(
    muh_draw + (lk_xPPM ? dot(view(Xlk_covariates,j,:,t), beta_iter[t]) : 0),
    sqrt(sig2h_draw)),
    Y[j,t]) + lPP[1]
```

and through in-place operations

```
copy!(s1n, @view sp1[aux_idxs])
copy!(s2n, @view sp2[aux_idxs])

spatial_cohesion!(spatial_cohesion_idx, s1n, s2n, sp_params_struct,
true, M_dp, S,1,true,lPP)
# LPP += spatial_cohesion!(spatial_cohesion_idx, s1n, s2n,
sp_params_struct, true, M_dp, S,1,true)
```

General optimizations

- benchmarking different possible solutions through the package `BenchmarkTools`

```
using BenchmarkTools
nh_tmp = rand(100)
@btime nclus_temp = sum($nh_tmp .> 0)
# 168.956 ns (2 allocations: 112 bytes)
@btime nclus_temp = count(x->(x>0), $nh_tmp)
# 11.612 ns (0 allocations: 0 bytes)
```

- refining the definition of cohesion and similarity functions, which are performance-critical since they are called thousands or even millions of times at every fit

Optimizing spatial cohesions

Problem: optimizing cohesions C_3 and C_4.
Solution v1: naive vector implementation.

```
sbar = [mean(s1), mean(s2)]
vtmp = sbar - mu_0
Mtmp = vtmp * vtmp'
Psi_n = Psi + S + (k0*sdim) / (k0+sdim) * Mtmp
⋮
```

Optimizing spatial cohesions

Problem: optimizing cohesions C_3 and C_4.
Solution v2: implementation using only scalar variables.

```
sbar1 = mean(s1)
sbar2 = mean(s2)
vtmp_1 = sbar1 - mu_0[1]
vtmp_2 = sbar2 - mu_0[2]
Mtmp_1 = vtmp_1^2
Mtmp_2 = vtmp_1 * vtmp_2
Mtmp_3 = copy(Mtmp_2)
Mtmp_4 = vtmp_2^2
aux1 = k0 * sdim; aux2 = k0 + sdim
Psi_n_1 = Psi[1] + S1 + aux1 / (aux2) * Mtmp_1
Psi_n_2 = Psi[2] + S2 + aux1 / (aux2) * Mtmp_2
Psi_n_3 = Psi[3] + S3 + aux1 / (aux2) * Mtmp_3
Psi_n_4 = Psi[4] + S4 + aux1 / (aux2) * Mtmp_4
⋮
```

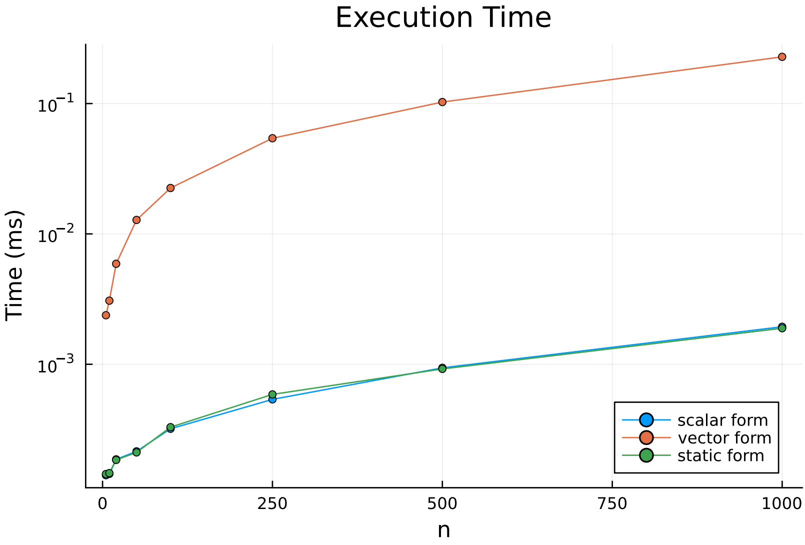
Optimizing spatial cohesions

Problem: optimizing cohesions C_3 and C_4.

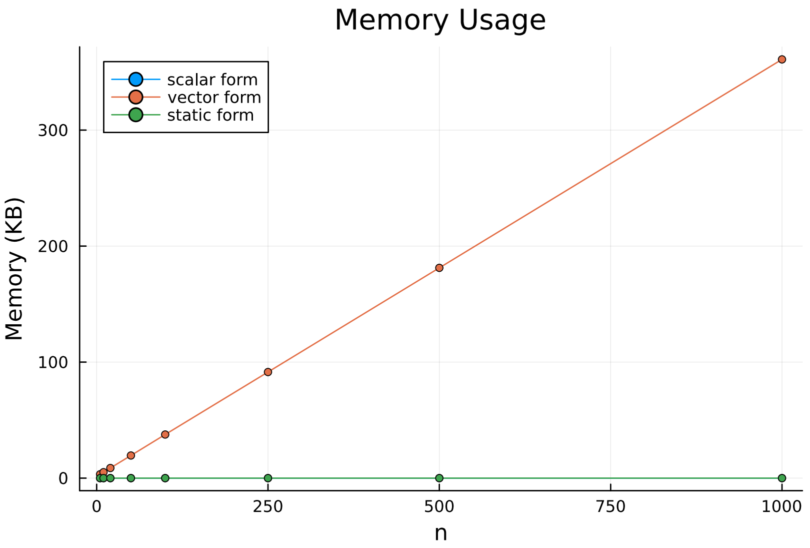
Solution: implementation using static vectors and matrices.

```
using StaticArrays
sbar1 = mean(s1); sbar2 = mean(s2)
sbar = SVector((sbar1, sbar2))
vtmp = sbar .- mu_0
Mtmp = vtmp * vtmp'
aux1 = k0 * sdim; aux2 = k0 + sdim
Psi_n = Psi .+ S .+ aux1 / (aux2) .* Mtmp
⋮
```

Optimizing spatial cohesions



Optimizing spatial cohesions

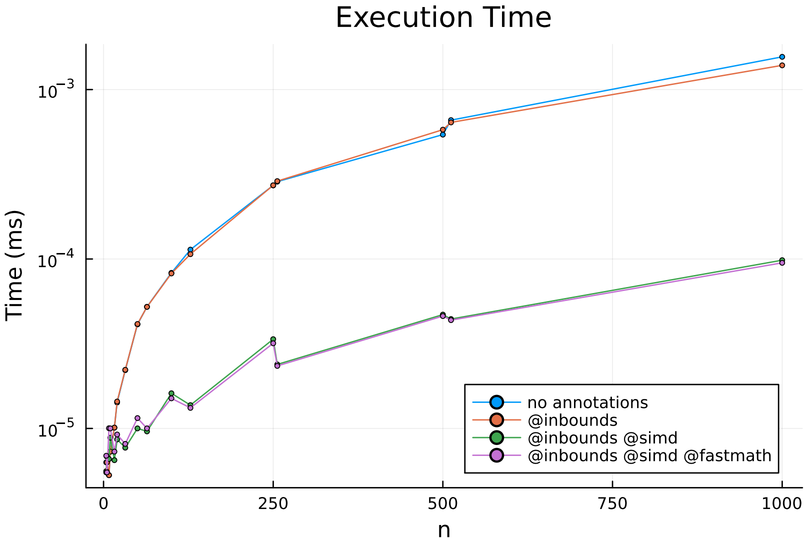


Optimizing covariates similarities

Problem: optimizing similarity g_4.
Solution: employing some optimizing macros on the inner loop.

```
function similarity4(X_jt::AbstractVector{<:Real}, mu_c::Real,
↪   lambda_c::Real, a_c::Real, b_c::Real, lg::Bool)
    n = length(X_jt); nm = n/2
    xbar = mean(X_jt)
    aux2 = 0.
    @inbounds @simd for i in eachindex(X_jt)
        aux2 += X_jt[i]^2
    end
    aux1 = b_c + 0.5 * (aux2 - (n*xbar + lambda_c*mu_c)^2/(n+lambda_c) +
↪   lambda_c*mu_c^2 )
    out = -nm*log2pi + 0.5*log(lambda_c/(lambda_c+n)) + lgamma(a_c+nm) -
↪   lgamma(a_c) + a_c*log(b_c) + (-a_c-nm)*log(aux1)
    return lg ? out : exp(out)
end
```

Optimizing covariates similarities



Comparing the two algorithms

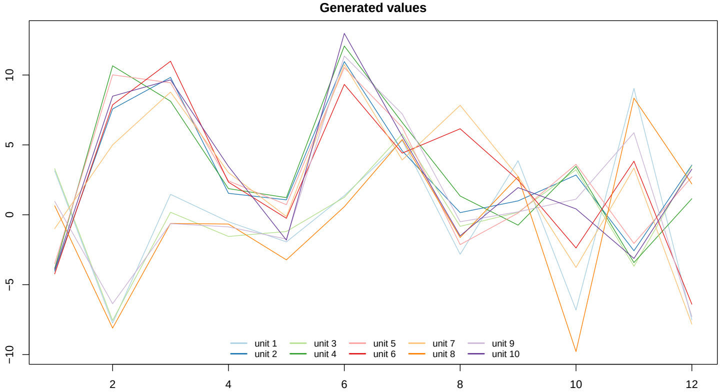
Our model represents a generalization of the original DRPM and its associated MCMC algorithm. Therefore, under identical datasets, hyperparameters, and MCMC configurations, both models are expected to perform similarly and produce comparable clusters estimates.

To evaluate the numerical performance of both algorithms, we will analyse posterior samples and clusters estimates in two scenarios:

1. using a synthetic dataset that includes only the response variable
2. employing a real-world spatio-temporal dataset, derived from the AgrImOnIA project (**agrimonia**)

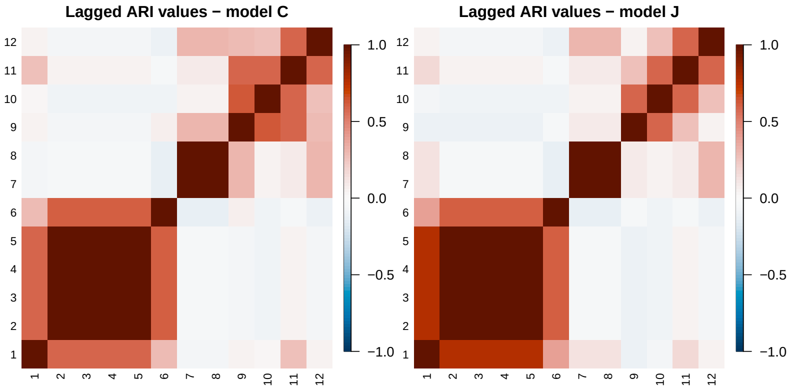
Simulated data scenario

For the first comparison we generated a dataset consisting of $n = 10$ units and $T = 12$ time instants. Both algorithms were executed collecting 2000 iterates from a total of 50000 iterations, by discarding the first 40000 as burnin and then thinning by 5.



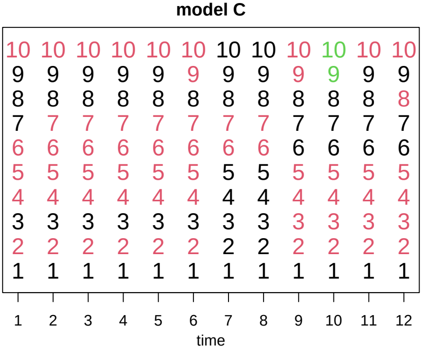
Simulated data scenario

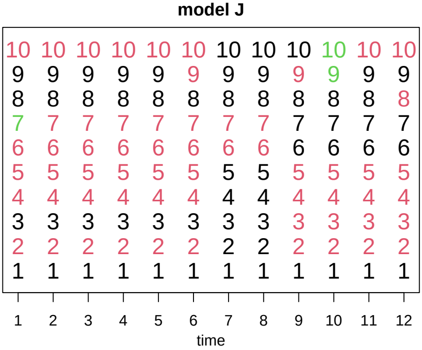
	MSE mean	MSE median	execution time
CDRPM	1.6221	1.5823	19s
JDRPM	**1.2634**	**1.2034**	**13s**



Simulated data scenario

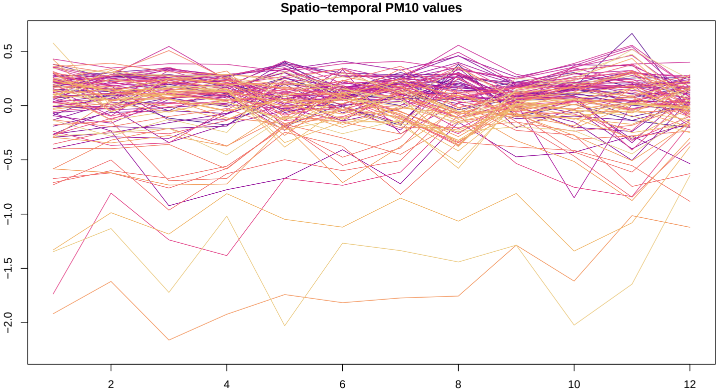
Computing the adjusted Rand index $\mathrm{ARI}(\rho_{\mathrm{JDRPM}}(t), \rho_{\mathrm{CDRPM}}(t))$ for all time instants $t = 1, \ldots, 12$, we obtained a mean of 0.95 and a median of 1.





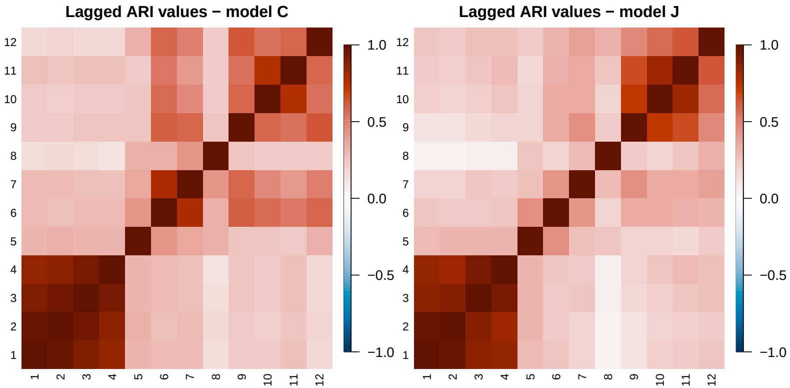
Real-world scenario

For the second comparison, we employed a dataset comprising weekly averages of PM_{10} values from the year 2018, measured by $n = 105$ units for $T = 12$ time instants. We collected 4000 iterates from 110000 total iterations, by discarding the first 90000 as burnin and then thinning by 5.



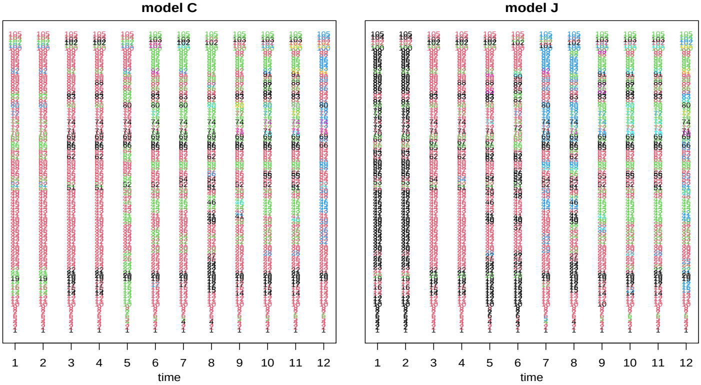
Real-world scenario

	MSE mean	MSE median	execution time
CDRPM	0.0142	0.0149	1h38m
JDRPM	**0.0131**	**0.0138**	**48m**



Real-world scenario

Computing the adjusted Rand index $\text{ARI}(\rho_{\text{JDRPM}}(t), \rho_{\text{CDRPM}}(t))$ for all time instants $t = 1, \ldots, 12$, we obtained a mean of 0.80 and a median of 0.86, denoting a strong agreement between the clusters estimates.



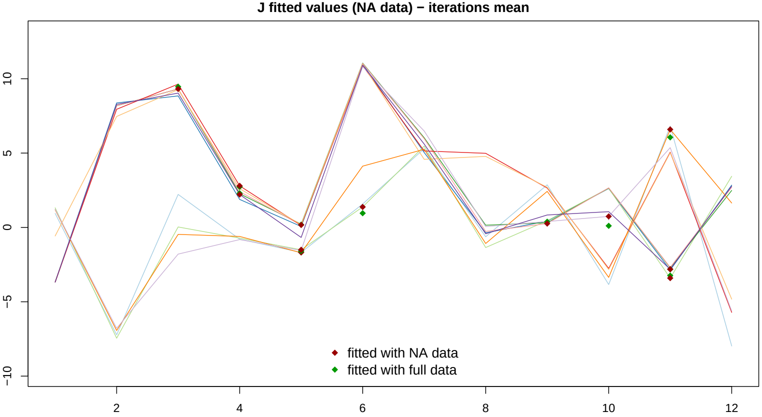
Performance with missing values

We then replicated the analyses focusing on scenarios involving missing values, with the objective of investigating how the JDRPM performs in the absence of complete datasets and to determine whether it maintains effective performances under such conditions.

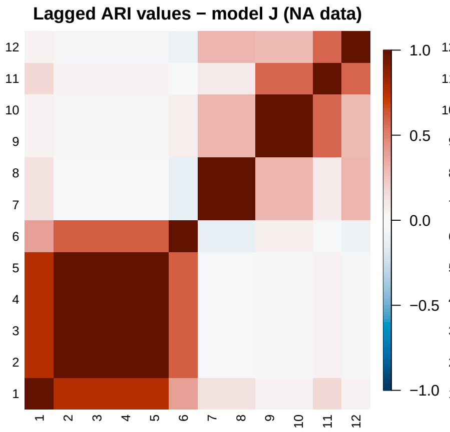
Given the extent of missing values in the AgrImOnIA dataset (**agrimonia**), which was used for the spatio-temporal analysis, we opted to set 10% of the values as missing (NAs). To implement this, we randomly selected $nT/10$ indexes from the sets $[1, \dots, n]$ and $[1, \dots, T]$ to identify all the pairs (i, t) that would be designated as missing entries in the target variable Y_{it}.

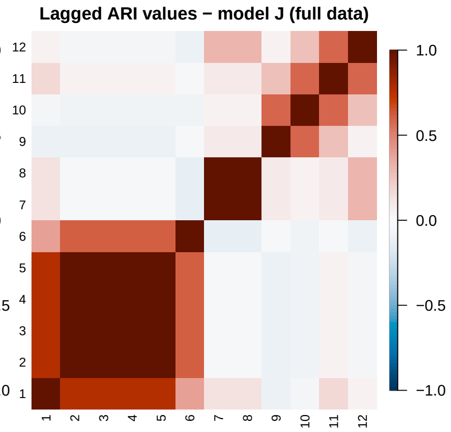
Simulated data

	MSE mean	MSE median	LPML	WAIC	exec. time
full data	1.2634	1.2034	-223.36	393.97	13s
NA data	1.4721	1.2101	-236.93	401.44	13s

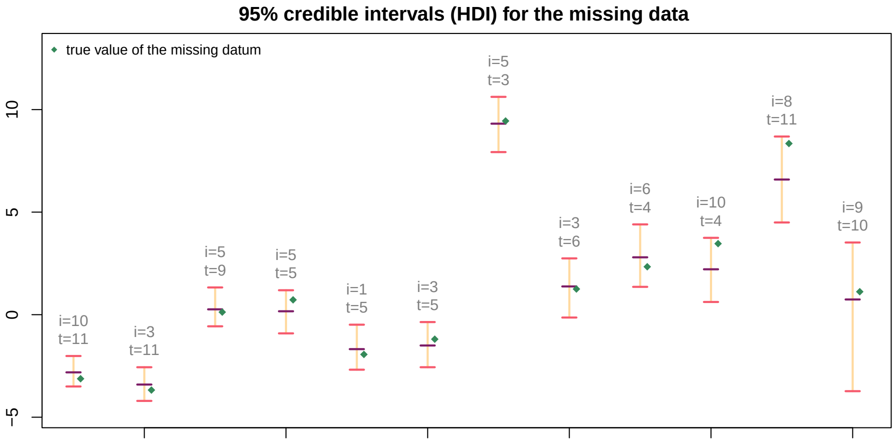


Simulated data



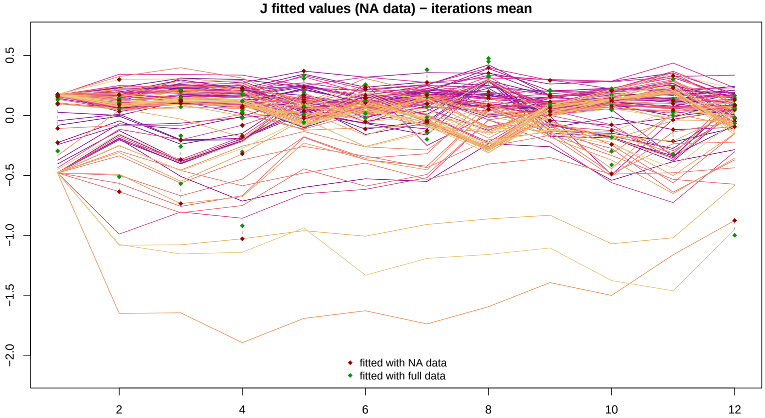


Simulated data

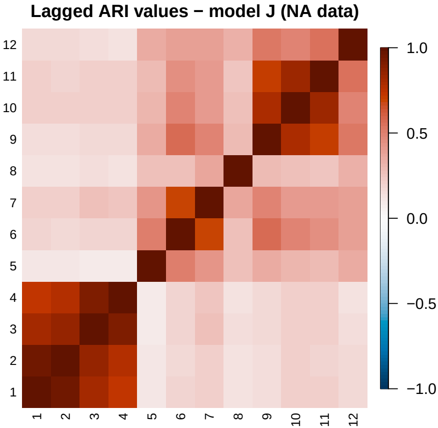


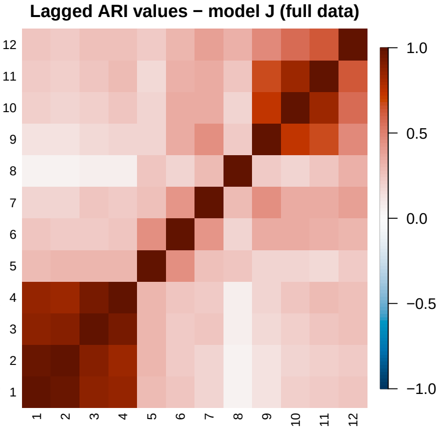
Real-world scenario

	MSE mean	MSE median	LPML	WAIC	exec. time
full data	0.0131	0.0138	624.91	-1898.05	48m
NA data	0.0160	0.0170	502.86	-1793.64	43m



Real-world scenario





Real-world scenario

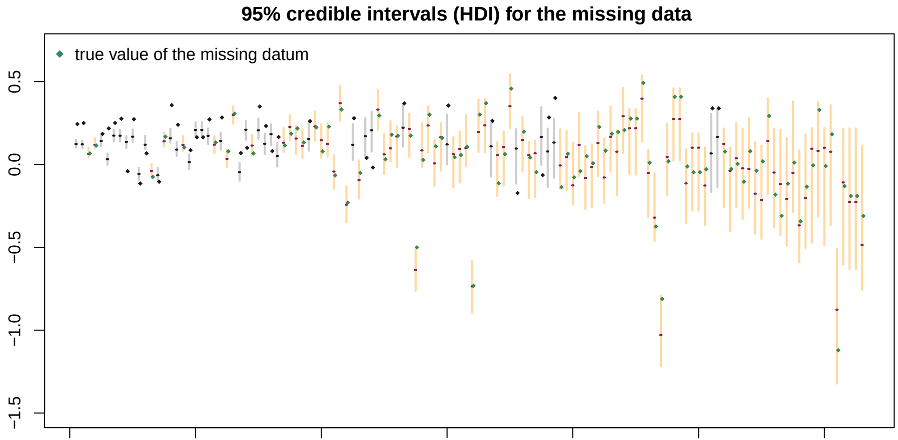


Figure: 74% of the true values lie within the credible intervals.

Effects of the covariates

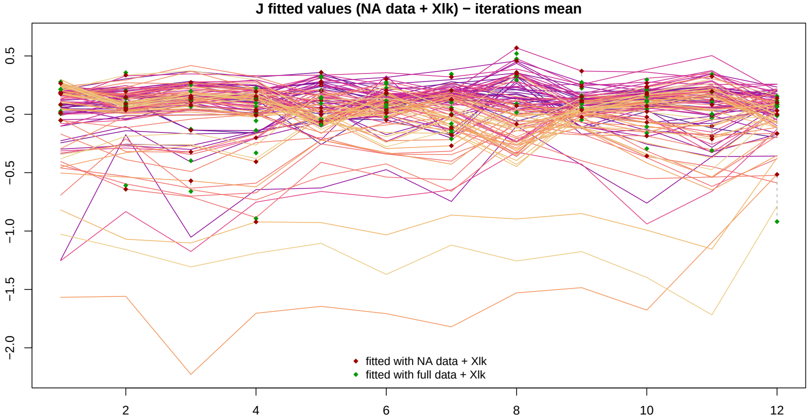
We then conducted several experiments to explore the key advancement introduced by the JDRPM: the inclusion of covariates.

Given their distinctly different purposes, we studied separately their effects:

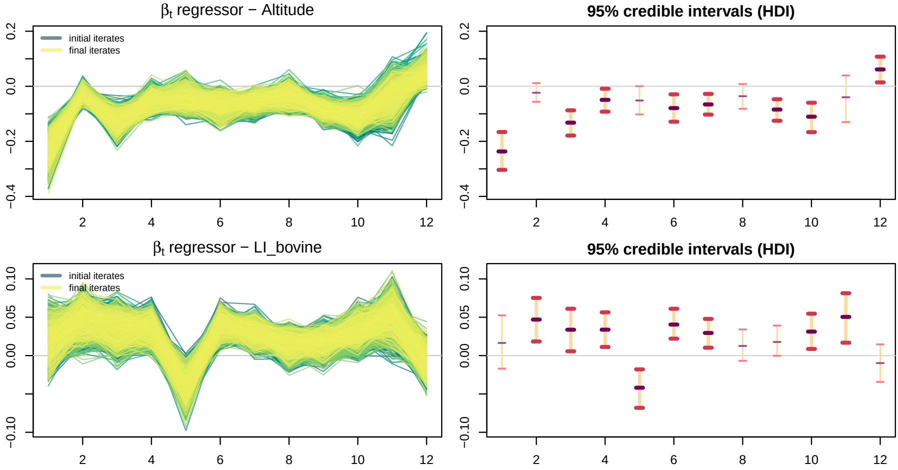
- covariates in the likelihood are expected to improve the estimation quality of the target variable Y_{it} and the associated model parameters $\implies$ context of the spatio-temporal experiments, with missing data
- covariates in the prior are expected to improve the accuracy and interpretability clusters estimates $\implies$ context of the spatio-temporal experiments

Covariates in the likelihood

	MSE mean	MSE median	LPML	WAIC	exec. time
full data	0.0131	0.0138	624.91	-1898.05	48m
NA data	0.0160	0.0170	502.86	-1793.64	**43m**
NA data + XIk	**0.0127**	**0.0130**	**625.81**	**-1902.74**	58m

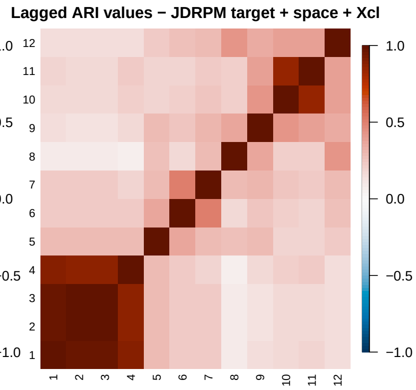


Covariates in the likelihood



Covariates in the prior

	MSE mean	MSE median	LPML	WAIC	exec. time
CDRPM	0.0142	0.0149	694.81	-1768.42	1h38m
JDRPM	0.0131	0.0138	624.91	-1898.05	**48m**
JDRPM + Xcl	**0.0126**	**0.0135**	**677.71**	**-1969.76**	1h20m



Covariates in the prior

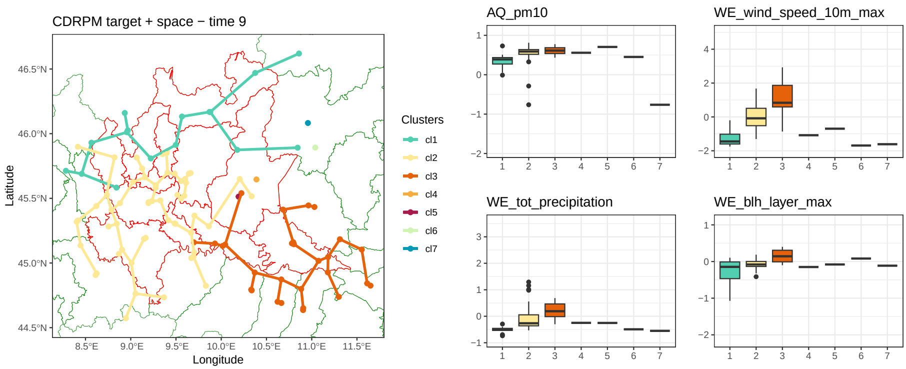


Figure: CDRPM spatially-informed fit.

Covariates in the prior

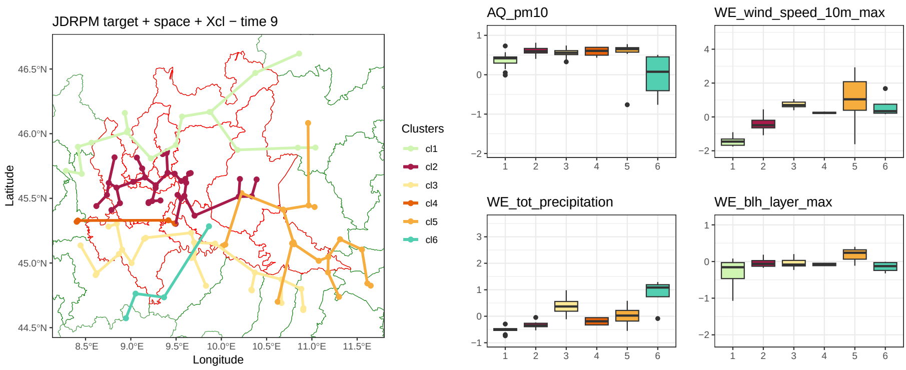
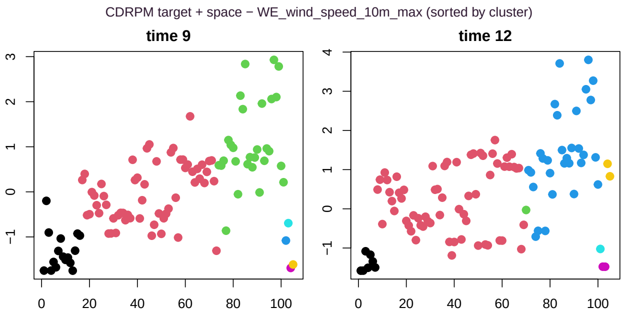
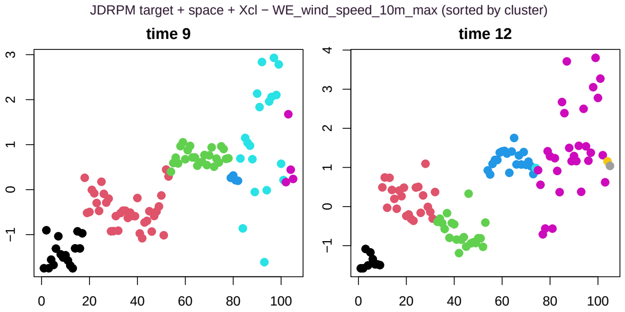


Figure: JDRPM spatially-informed fit with covariates in the prior.


CDRPM target + space − WE_wind_speed_10m_max (sorted by cluster)
time 9
time 12

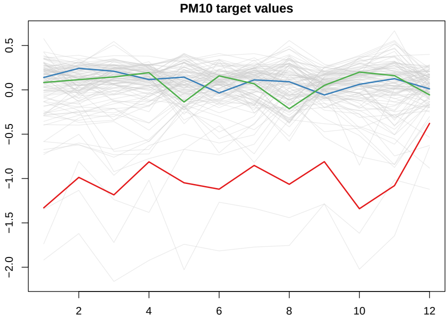

JDRPM target + space + Xcl − WE_wind_speed_10m_max (sorted by cluster)
time 9
time 12

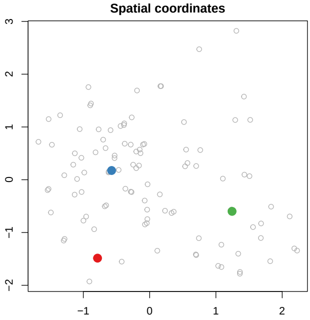
Inference on a new location

As a final experiment on the effects of covariates, we considered a spatio-temporal scenario in which new units were added at new locations, with the objective of inferring the values of their target variable time series. We reproduced this scenario by removing all data entries from three randomly-selected units within the spatio-temporal dataset.

This context resembles the real use-case of predicting the behaviour of a unit for which sensors may be absent or inactive, with the expectation that the estimation accuracy will improve as model complexity increases.

		space	space+Xlk	space+Xlk+Xcl
unit 92 (red)	MSE mean	0.112452	**0.042037**	0.044957
	MSE median	0.111573	**0.041676**	0.045216
unit 61 (blue)	MSE mean	0.004117	**0.002449**	0.002527
	MSE median	0.004711	0.002547	**0.002534**
unit 44 (green)	MSE mean	**0.003919**	0.006368	0.005945
	MSE median	**0.003997**	0.006419	0.005950





Inference on a new location

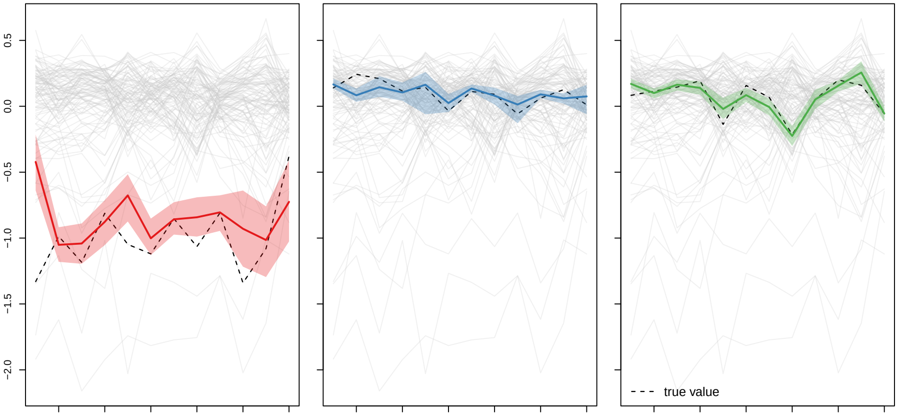


Figure: JDRPM spatially-informed fit.

Inference on a new location

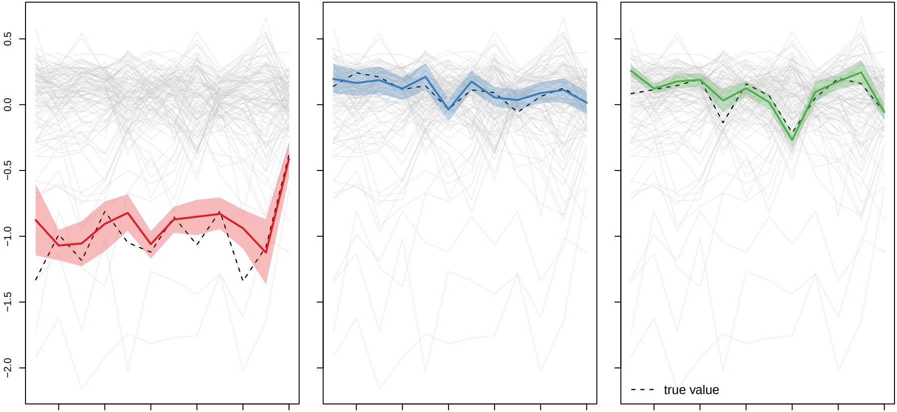


Figure: JDRPM spatially-informed fit with covariates in the likelihood.

Inference on a new location

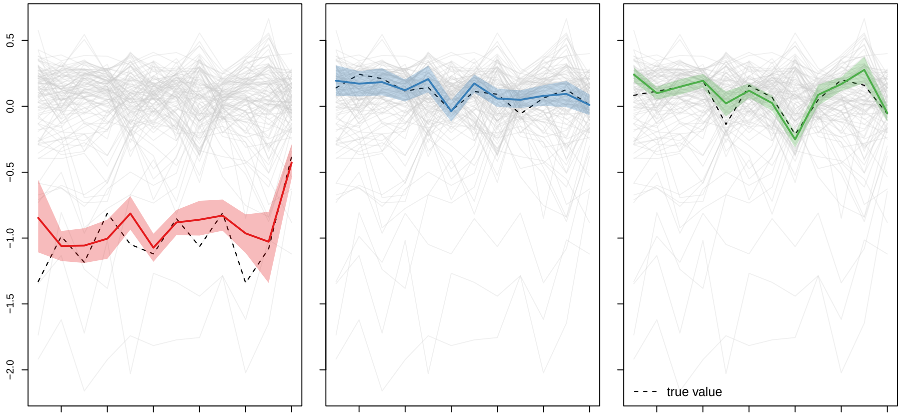


Figure: JDRPM spatially-informed fit with covariates in the likelihood and in the prior.

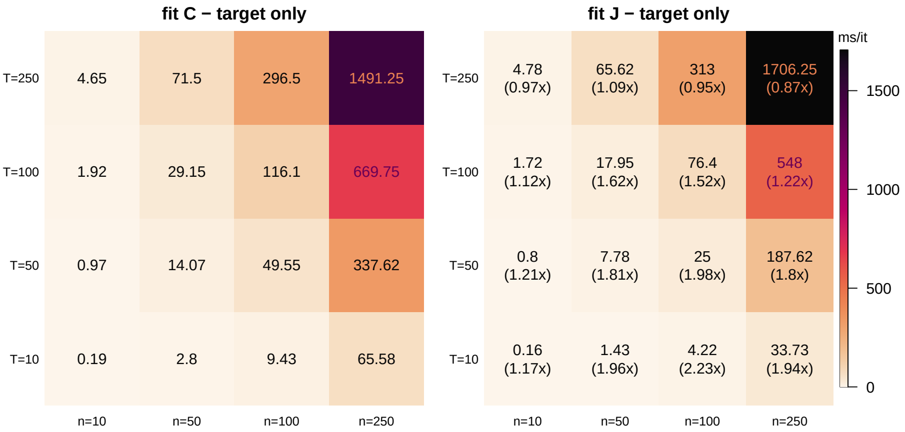
Scaling performances

Finally, we designed a set of experiments to evaluate the computaional performances of CDRPM's and JDRPM's implemntations.

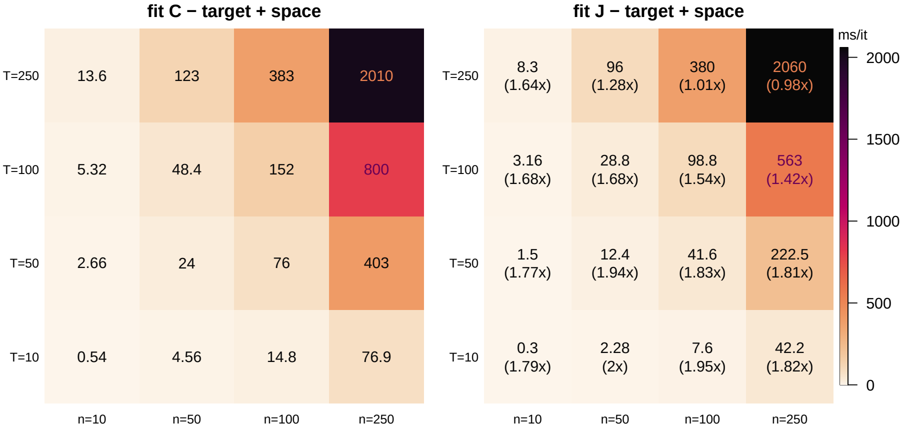
We fitted both models across a "mesh" of dataset sizes, with both the number of units n and the time horizons T ranging through the set $\{10, 50, 100, 250\}$, and with information layers inserted incrementally on top of each other.

To measure the average execution time per iteration of each fit we defined the number of iterations to be inversely proportional to the size of the dataset, and repeated each fit was repeated multiple times to record the minimum execution time observed.

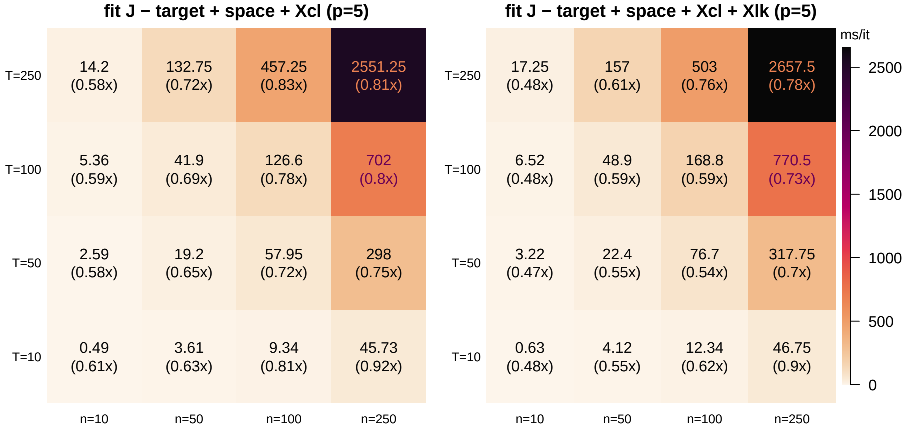
Performances in the simulated data scenario



Performances in the real-world scenario

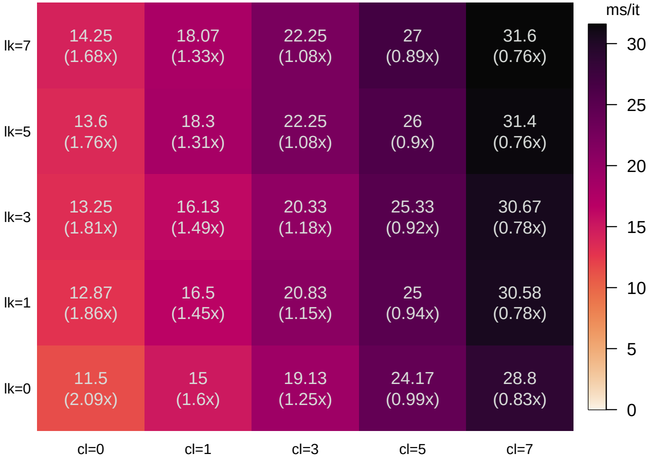


Performances - varying n and T, fixed p_{lk} and p_{cl}

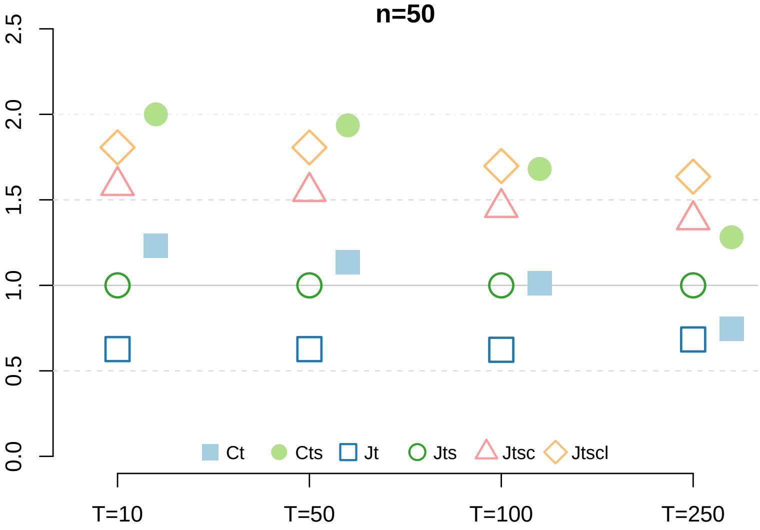


Performances - fixed n and T, varying p_{lk} and p_{cl}

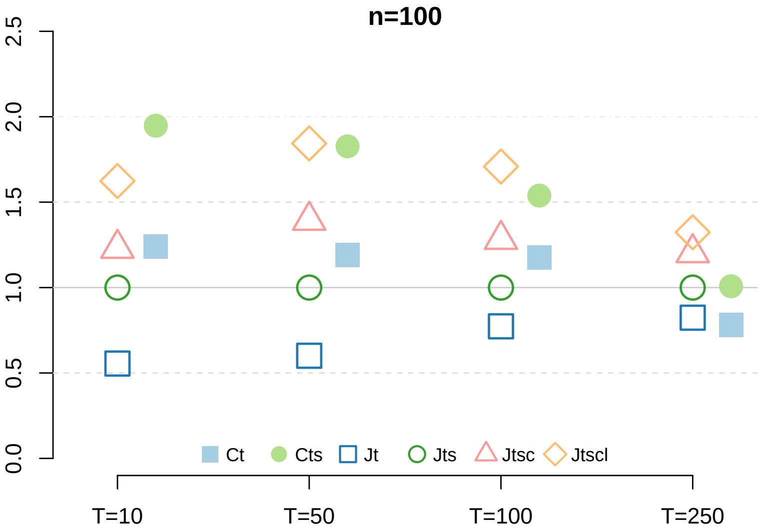
fit J − space + Xlk + Xcl for varying p



Summary performances



Summary performances



Strengths

- JDRPM retains the foundational structure of its predecessor CDRPM, with the temporal modelling of the sequence of partitions
- the introduction of covariates information, as well as the accommodation of missing values, should allow more flexibility in the real-world researches
- despite the increased complexity, we provided more efficiency in the implementation, significantly reducing execution times
- the choice of the Julia language should facilitate easier code developments and future variations

Drawbacks

- the robustness of the fits may decrease due to the intricacies of parameters selection both in the prior distributions as well as in the cohesion and similarity functions
- reaching an appropriate balance between spatial and covariates information may require empirical testing (however, to address this problem, the Julia code already provides an optional argument, `cv_weight`, defaulted to 1, that allows to adjust the influence of covariates similarities)
- the choice of the inverse gamma distribution as the prior of the variance parameters allows better mixing properties but is more delicate to tune, compared to a simpler uniform distribution


That's all Folks!

Bibliography I