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 (Page et al., 2022) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

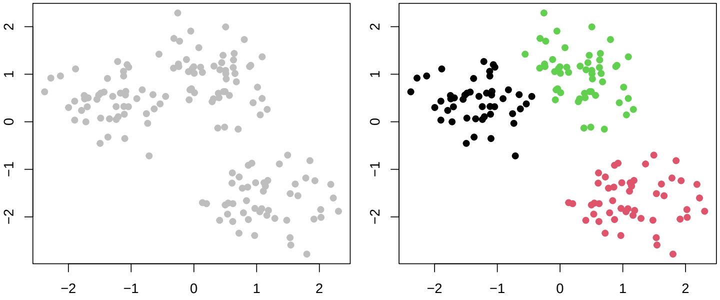
Currently, the model's implementation

- 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 in Julia rather than C

Clustering

The Dependent Random Partition Model from (Page et al., 2022) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

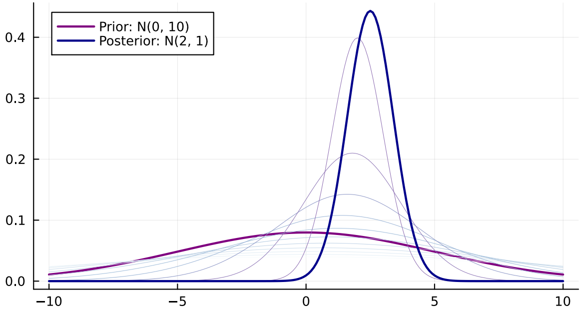
Clustering is a fundamental technique of unsupervised learning where a set of data points has to be divided into homogeneous groups of units which exhibit a similar behaviour.



Why going Bayesian?

The Dependent Random Partition Model from (Page et al., 2022) is a Bayesian spatio-temporal clustering model which directly models the temporal dependencies in the sequence of clusters over time.

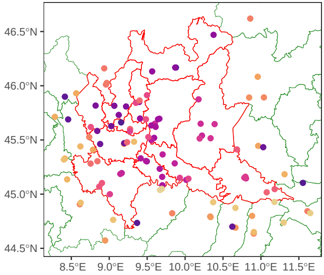
Bayesian models incorporate prior information on the model parameters and allow to assess uncertainty when performing inference on the results.

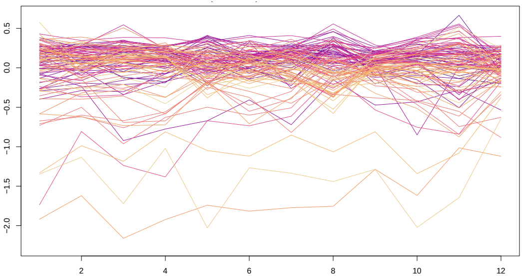


A bit of (spatio-temporal) context

The Dependent Random Partition Model from (Page et al., 2022) 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 (Page et al., 2022) 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.

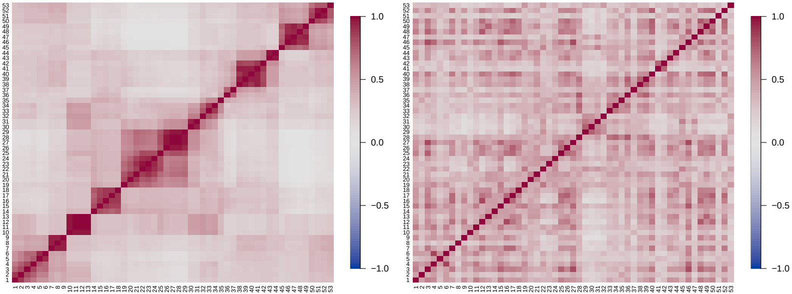


Figure: ARI($\hat{\rho}_t, \hat{\rho}_{t+k}$) computed for DRPM (left) and for a competitor model (right).

Why should we care about temporal dependencies?

The Dependent Random Partition Model from (Page et al., 2022) 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.

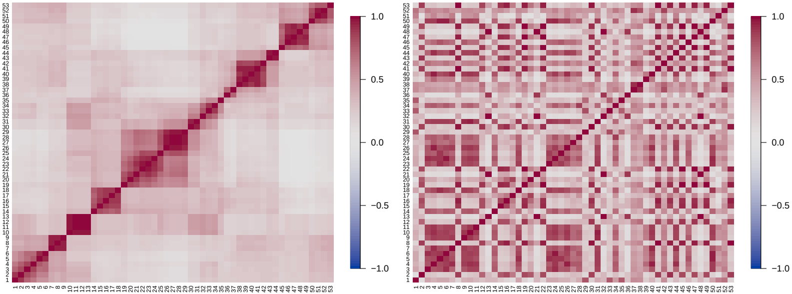


Figure: ARI($\hat{\rho}_t, \hat{\rho}_{t+k}$) computed for DRPM (left) and for a competitor model (right).

Modelling the temporal dependence (Page et al., 2022)

To introduce temporal dependence in a collection of partitions, Page et al. (2022) assumed a first-order Markovian structure, leading to

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

where $P(\rho_1)$ is an exchangeable partition probability function (EPPF). Page et al. (2022) chose the EPPF induced by the Dirichlet Process

$$P(\rho_1) \propto \prod_{j=1}^{k_1} M \cdot (|S_{j1}| - 1)!$$

Modelling the temporal dependence (Page et al., 2022)

To characterize the other terms $P(\rho_t|\rho_{t-1})$, the following auxiliary variables need to be introduced. For all units $i = 1, \dots, n$ 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
  $\implies$ 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
  $\implies$ their cluster configurations will change significantly
  $\implies$ the majority of γ_{it} will be 0

Modelling the temporal dependence (Page et al., 2022)

Page et al. (2022) 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 ($\alpha_t = 1$).

In this way the formulation of the joint model becomes

$$
\begin{aligned}
P(\boldsymbol{\gamma}_1, \rho_1, \ldots, \boldsymbol{\gamma}_T, \rho_T) = {} & P(\rho_T | \boldsymbol{\gamma}_T, \rho_{T-1}) P(\boldsymbol{\gamma}_T) \times \cdots \\
& \times P(\rho_2 | \boldsymbol{\gamma}_2, \rho_1) P(\boldsymbol{\gamma}_2) P(\rho_1)
\end{aligned}
$$

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

The DRPM (Page et al., 2022)

DRPM formulation according to Page et al. (2022) (henceforth, CDRPM, with C as the C language used for the model's implementation).

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

Our generalized model

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

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

Our updated formulation

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

$$Y_{it} | Y_{it-1}, \boldsymbol{\mu}_t^\star, \boldsymbol{\sigma}_t^{2\star}, \boldsymbol{\eta}, \boldsymbol{c}_t \overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{it}t} + \eta_{1i} Y_{it-1} + \boldsymbol{x}_{it}^T \beta_t, \sigma^{2\star}_{c_{it}t}(1 - \eta_{1i}^2))$$

$$Y_{i1} \overset{\text{ind}}{\sim} \mathcal{N}(\mu^\star_{c_{i1}1} + \boldsymbol{x}_{i1}^T \beta_1, \sigma^{2\star}_{c_{i1}1})$$

$$\beta_t \overset{\text{ind}}{\sim} \mathcal{N}_p(\boldsymbol{b}, s^2 I)$$

$$\xi_i = \text{Logit}(\tfrac{1}{2}(\eta_{1i} + 1)) \overset{\text{ind}}{\sim} \text{Laplace}(a, b)$$

$$(\mu^\star_{jt}, \sigma^{2\star}_{jt}) \overset{\text{ind}}{\sim} \mathcal{N}(\vartheta_t, \tau_t^2) \times \text{invGamma}(a_\sigma, b_\sigma)$$

$$\vartheta_t | \vartheta_{t-1} \overset{\text{ind}}{\sim} \mathcal{N}((1 - \varphi_1)\varphi_0 + \varphi_1 \vartheta_{t-1}, \lambda^2(1 - \varphi_1^2))$$

$$(\vartheta_1, \tau_t^2) \overset{\text{iid}}{\sim} \mathcal{N}(\varphi_0, \lambda^2) \times \text{invGamma}(a_\tau, b_\tau)$$

$$(\varphi_0, \varphi_1, \lambda^2) \sim \mathcal{N}(m_0, s_0^2) \times \mathcal{U}(-1, 1) \times \text{invGamma}(a_\lambda, b_\lambda)$$

$$\{\boldsymbol{c}_t, \ldots, \boldsymbol{c}_T\} \sim \text{tRPM}(\boldsymbol{\alpha}, M) \text{ with } \alpha_t \overset{\text{iid}}{\sim} \text{Beta}(a_\alpha, b_\alpha)$$

Our updated formulation

The regressor term $\boldsymbol{\beta}_t$ provides more flexibility to the model formulation through the insertion of covariates into the likelihood.

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

Our updated formulation

The choice of the inverse gamma distribution for the variance parameters recovers conjugacy within the model.

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

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

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

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

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

Additional information level

2) We introduced covariates information inside the prior for the partition.

Page et al. (2022) introduced spatial information by moving to a PPM structure with a *cohesion function* $C(S_{jt}, \boldsymbol{s}^{\star}_{jt} | M, \mathcal{S})$, which measures the compactness of the spatial coordinates $\boldsymbol{s}^{\star}_{jt}$.

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

We implemented the spatial cohesion functions proposed in Page et al. (2016).

Additional information level

2) We introduced covariates information inside the prior for the partition.

In a similar way, we inserted covariates information by introducing a term composed by a *similarity function* $g(S_{jt}, \boldsymbol{x}^{\star}_{jtr}|\mathcal{C})$, which measures the similarity of the r-th covariate values $\boldsymbol{x}^{\star}_{jtr}$.

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

We implemented the covariates similarity functions proposed in Page et al. (2018).

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

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

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

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.

original MCMC algorithm of Page et al. (2022) → new MCMC algorithm

our updates → new MCMC algorithm

new MCMC algorithm → julia

Why Julia?

- combines the ease and expressiveness of high-level languages (e.g. R, python, matlab) with the efficiency and performance of low-level languages (e.g. C, C++, Fortran)
- 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
- vast collection of optimized and complete scientific packages, e.g. `Statistics`, `Distributions` (Besançon et al., 2021), and `BenchmarkTools` (Chen et al., 2016)

Spoiler alert

Using Julia, we obtained improved computational performance compared to the original C implementation of Page et al. (2022), with peaks up to a 2x speedup.

This performance gain was achieved through several optimizations steps.

Optimizing covariates similarities

Problem: optimizing covariates similarity g_4.

```
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.
    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
```

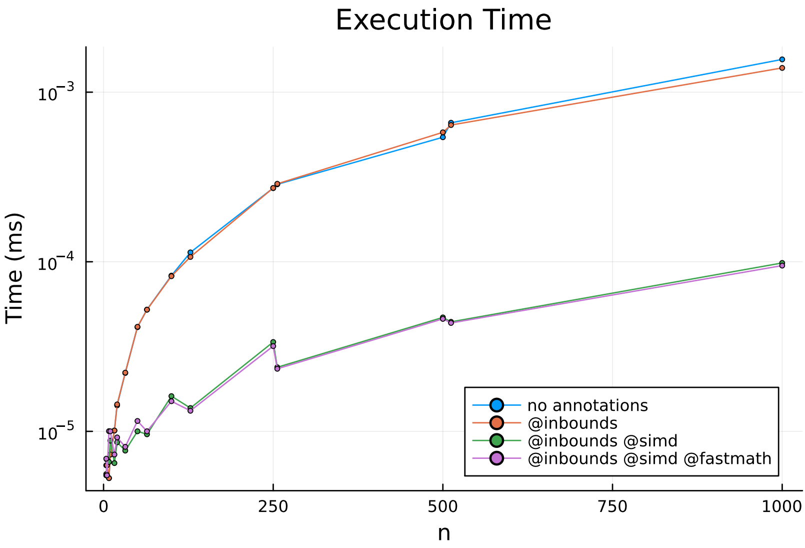
Optimizing covariates similarities

Problem: optimizing covariates similarity g_4.

Solution: applying 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 @fastmath @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



Analysis of the models

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

1. on a synthetic dataset, with kind-of-randomly generated data, that includes only the response variable
2. on a real-world spatio-temporal dataset about air pollution in Lombardy, derived from the AgrImOnIA project (Fassò et al., 2023)

We will start with an assessment question about our implementation, followed by a specific analysis of the upgrades brought by our work.

Correctness check

Is our generalized model working as expected?

We fitted both the original and our updated models under identical conditions: same datasets, same MCMC setup (burnin, thinning, number of iterations), and same hyperparameters.

Correctness check

Is our generalized model working as expected? *yes!*

Everything works nicely, JDRPM produces similar results to CDRPM.

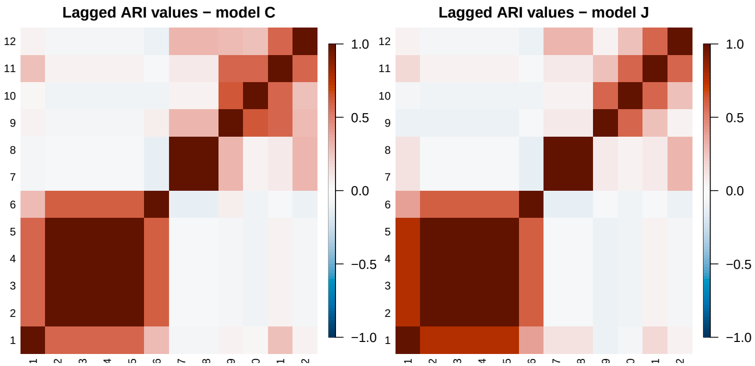


Figure: Lagged ARI values of the two models, fitted on simulated data.

Correctness check

Is our generalized model working as expected? *yes!*

Everything works nicely, JDRPM produces similar results to CDRPM.

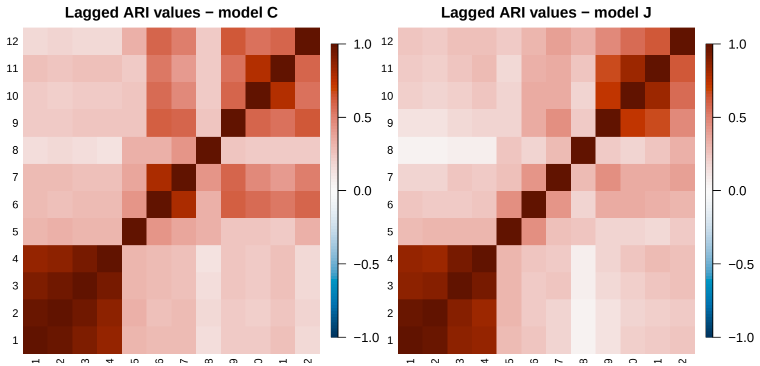


Figure: Lagged ARI values of the two models, fitted on real-world data.

Correctness check

Is our generalized model working as expected? *yes!*

Under identical testing conditions, the models are morally equivalent.

. . . *but our implementation is faster* 😏

simulated data	MSE mean	MSE median	execution time
CDRPM	1.6221	1.5823	19s (3.8 ms/it)
JDRPM	**1.2634**	**1.2034**	**13s (2.6 ms/it)**

real-world data	MSE mean	MSE median	execution time
CDRPM	0.0142	0.0149	1h38m (54 ms/it)
JDRPM	**0.0131**	**0.0138**	**48m (26 ms/it)**

Performance with missing values

How will our generalized model perform in presence of missing values?

We randomly removed 10% of the target values Y_{it} from the dataset, so that they would become our “missing values”, and then repeated the fits in both scenarios.

Performance with missing values

How will our generalized model perform in presence of missing values?

very well! All the true values lie within the credible intervals.

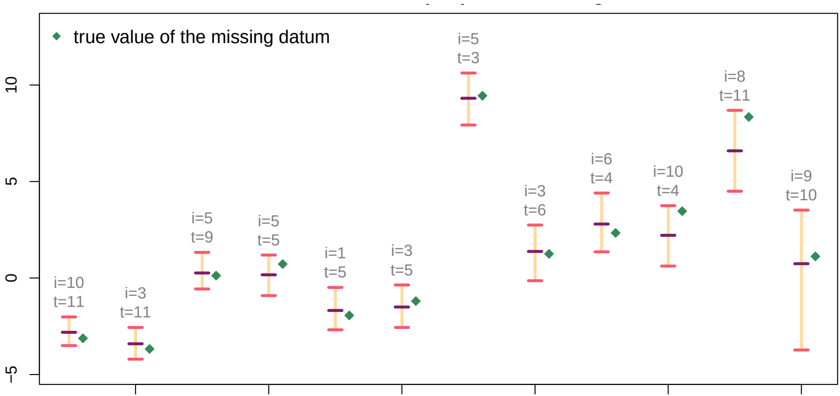


Figure: 95% CIs of the fitted estimates for the missing Y_{it}'s, on simulated data.

Performance with missing values

How will our generalized model perform in presence of missing values?

quite well! 74% of the true values lie within the credible intervals.

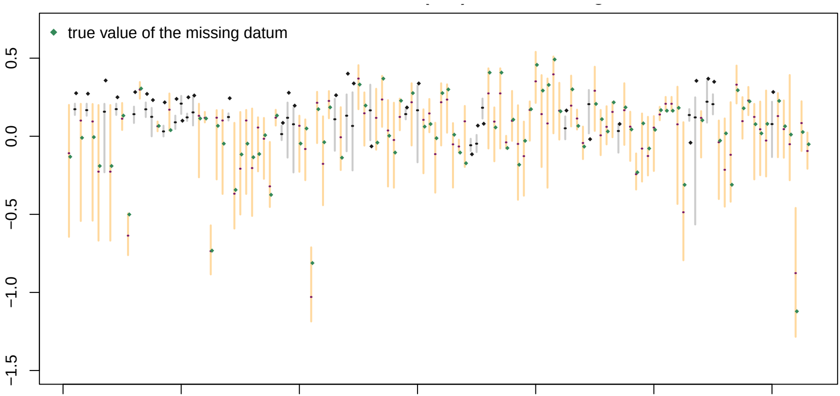


Figure: 95% CIs of the fitted estimates for the missing Y_{it}'s, on real-world data.

Performance with missing values

How will our generalized model perform in presence of missing values?

Also the temporal trend and the clusters estimates remain very similar.

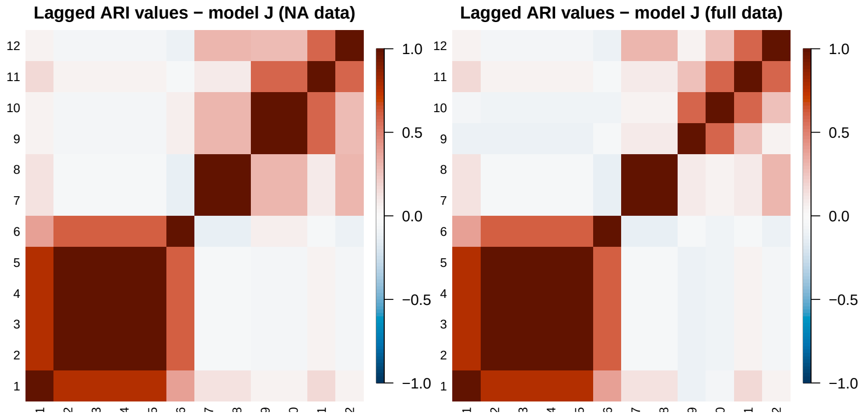


Figure: Lagged ARI values of the two JDRPM fits, on simulated data.

Performance with missing values

How will our generalized model perform in presence of missing values?

Also the temporal trend and the clusters estimates remain very similar.

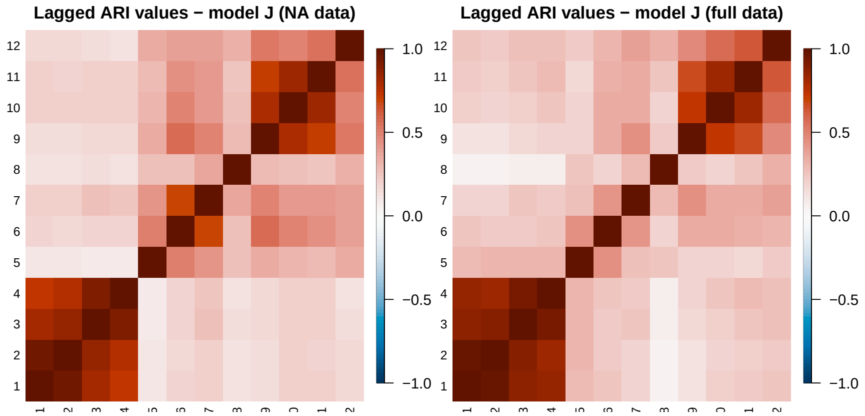


Figure: Lagged ARI values of the two JDRPM fits, on real-world data.

Covariates in the likelihood

What is the effect of including covariates in the likelihood?

We repeated the missing values analysis, this time including covariates in the likelihood.

Covariates in the likelihood

What is the effect of including covariates in the likelihood?

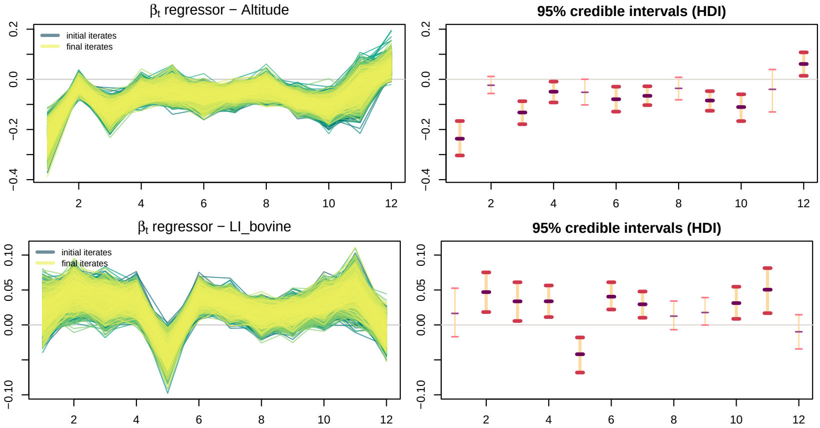
They improve the fitted estimates of Y_{it} and of the model parameters.

(JDRPM)	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 + Xlk	**0.0127**	**0.0130**	**625.81**	**-1902.74**	58m

Covariates in the likelihood

What is the effect of including covariates in the likelihood?

They provide insights about the included covariates.



Covariates in the prior

What is the effect of including covariates in the prior?

We repeated the real-world analysis, this time including covariates in the prior.

Covariates in the prior

What is the effect of including covariates in the prior?

They improve the quality and interpretability of the results.

	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

What is the effect of including covariates in the prior?

They improve the quality and interpretability of the results.

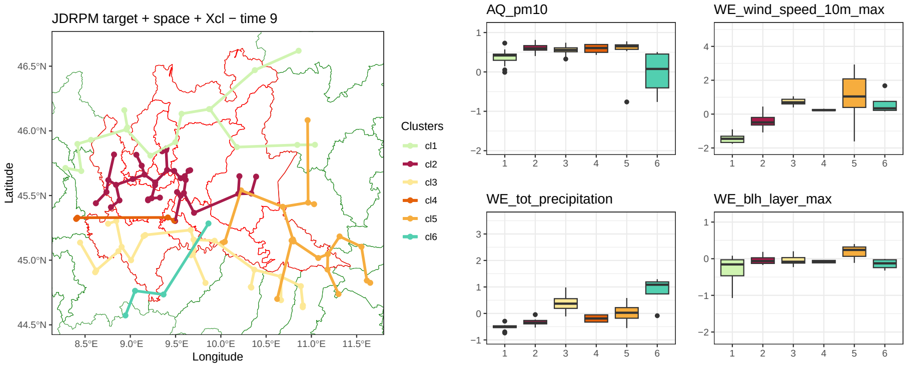


Figure: JDRPM spatially-informed fit with covariates in the prior.

Covariates in the prior

What is the effect of including covariates in the prior?

They improve the quality and interpretability of the results.

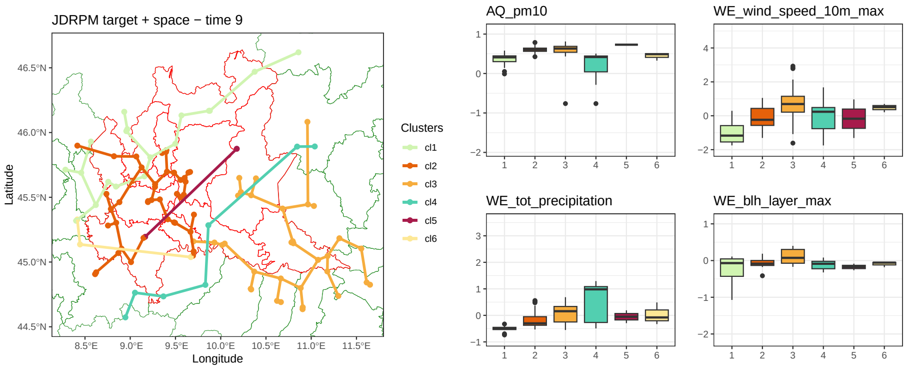


Figure: JDRPM spatially-informed fit.

Covariates in the prior

What is the effect of including covariates in the prior?

They improve the quality and interpretability of the results.

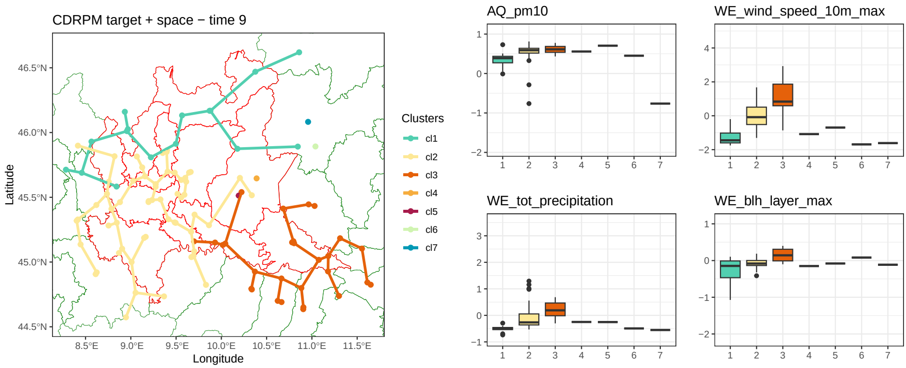


Figure: CDRPM spatially-informed fit.

Scaling performances

By how much is our implementation faster?

We fitted both models across a “mesh” of dataset sizes, with n and T ranging through $\{10, 50, 100, 250\}$ and with information layers inserted incrementally on top of each other.

Scaling performances

By how much is our implementation faster? *By a lot* 😎

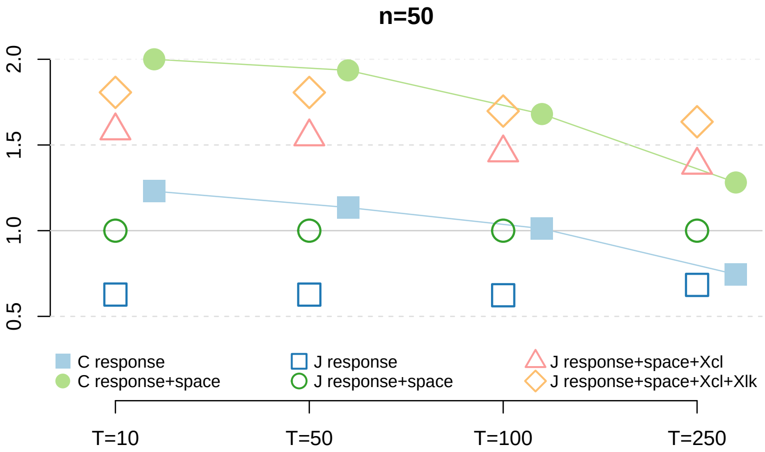


Figure: Slowdown factors computed for all fits, relative to JDPRM spatially-informed fit.

Scaling performances

By how much is our implementation faster? *Hardware is the limit*

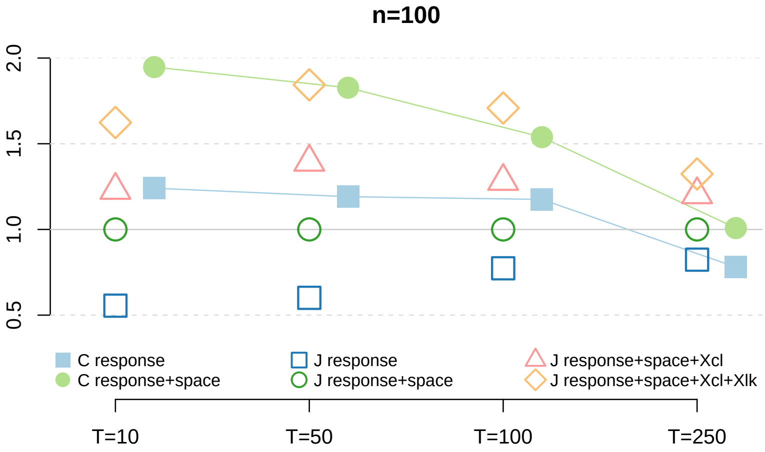


Figure: Slowdown factors computed for all fits, relative to JDPRM spatially-informed fit.

Strengths of our work

Our generalization of the DRPM model

- will allow more flexibility in the real-world researches, thanks to the introduction of covariates and the handling of missing values
- despite the increased complexity, provides more efficiency in the implementation, thus significantly reducing execution times
- is implemented in Julia, facilitating easier code variations and future developments
- is already ready-to-use on R
  (all the Julia and R codes of the thesis are available at `https://github.com/federicomor/Tesi/tree/main/src/JDRPM`)

Drawbacks of our work

- the estimates of the model parameters will be sensible to the choice of hyperparameters, covariates similarities, spatial cohesions, etc.
  However this is a characteristic of all complex Bayesian models.
- reaching an appropriate balance between spatial and covariates information may require empirical testing.
  However, the Julia implementation already provides an optional argument, `cv_weight` (defaulted to 1), which allows to scale the contribute of covariates similarities.


That's all Folks!

Bibliography I

- Besançon, Mathieu et al. (2021). "Distributions.jl: Definition and Modeling of Probability Distributions in the JuliaStats Ecosystem". In: *Journal of Statistical Software* 98.16, pp. 1–30. ISSN: 1548-7660. DOI: 10.18637/jss.v098.i16.
- Chen, Jiahao and Jarrett Revels (Aug. 2016). "Robust benchmarking in noisy environments". In: *arXiv e-prints*, arXiv:1608.04295.
- Fassò, A. et al. (2023). *AgrImOnIA: Open Access dataset correlating livestock and air quality in the Lombardy region, Italy (3.0.0)*. DOI: https://doi.org/10.5281/zenodo.7956006.
- Page, Garritt L. and Fernando A. Quintana (Sept. 2018). "Calibrating covariate informed product partition models". In: *Statistics and Computing* 28, pp. 1–23. DOI: 10.1007/s11222-017-9777-z.
- — (2016). "Spatial Product Partition Models". In: *Bayesian Analysis* 11.1, pp. 265–298. DOI: 10.1214/15-BA971.

Bibliography II

Page, Garritt L., Fernando A. Quintana, and David B. Dahl (2022). "Dependent Modeling of Temporal Sequences of Random Partitions". In: *Journal of Computational and Graphical Statistics* 31.2, pp. 614–627. DOI: 10.1080/10618600.2021.1987255.

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` (Chen et al., 2016).

```
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
⋮
```

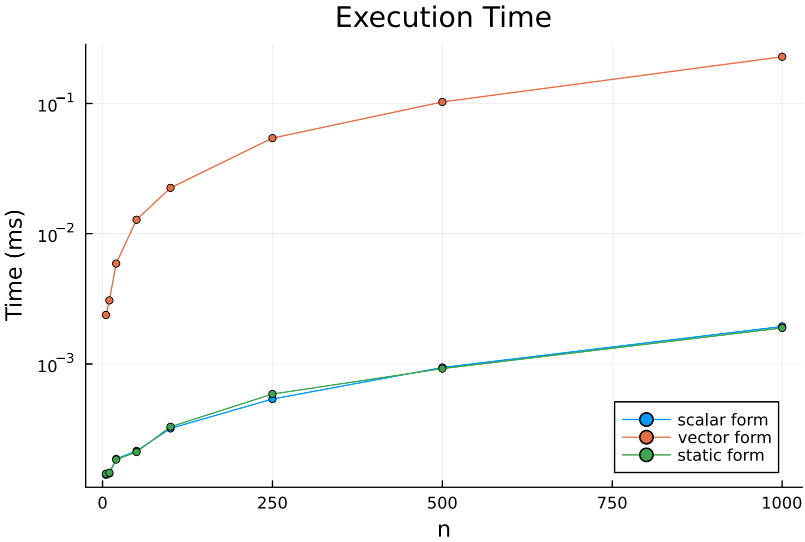
Optimizing spatial cohesions

Problem: optimizing cohesions C_3 and C_4.

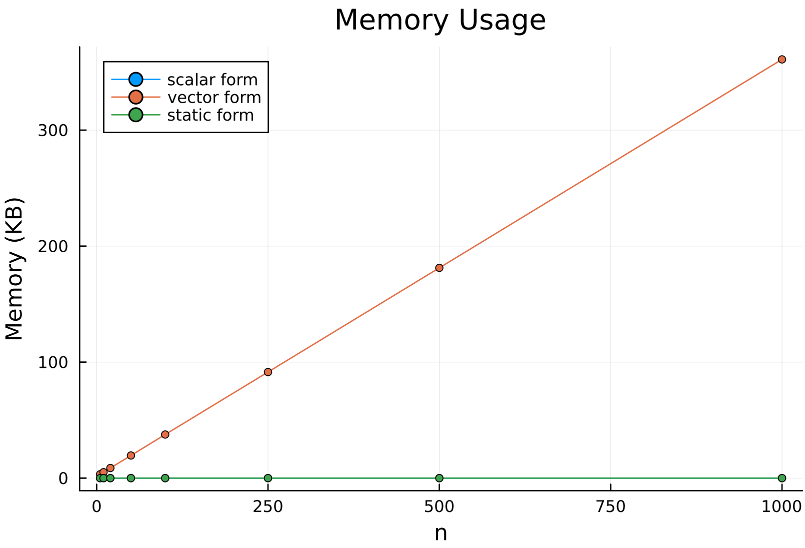
Solution: implementation using static vectors and matrices.

```
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

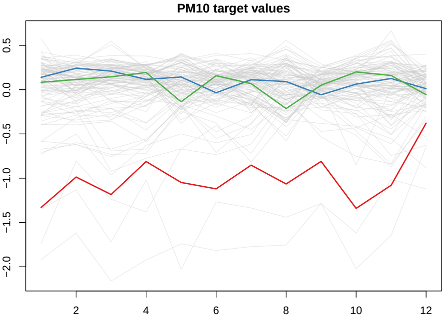


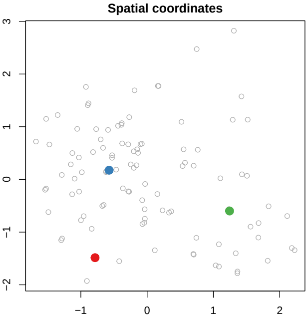
Inference on a new location

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

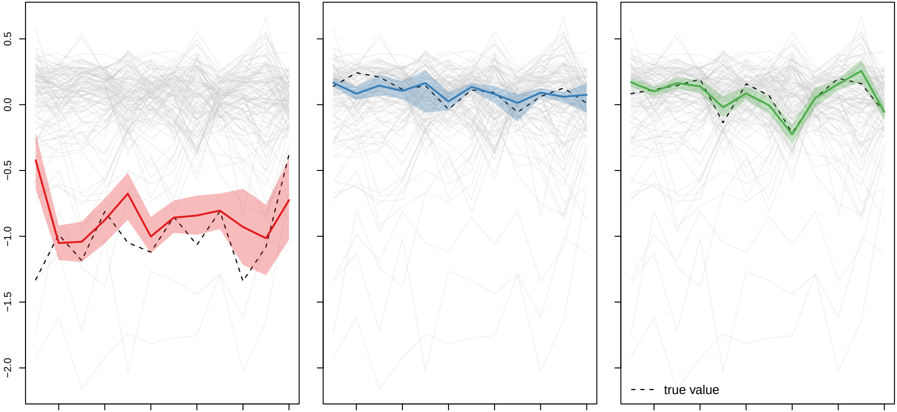


Figure: JDRPM spatially-informed fit.

Inference on a new location

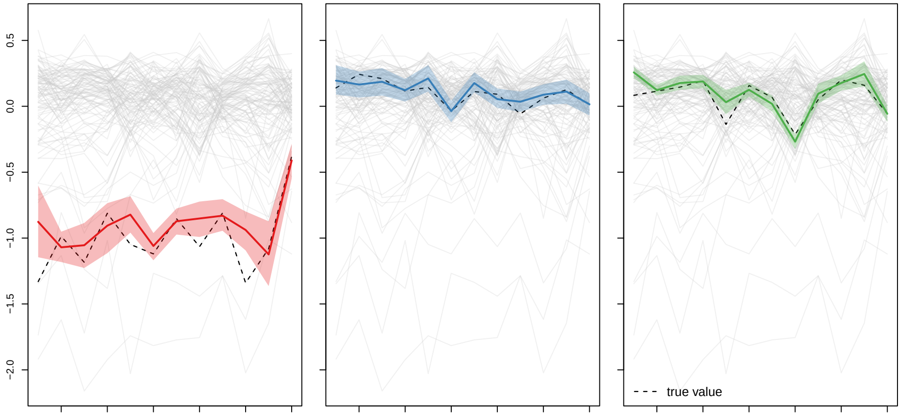


Figure: JDRPM spatially-informed fit with covariates in the likelihood.

Inference on a new location

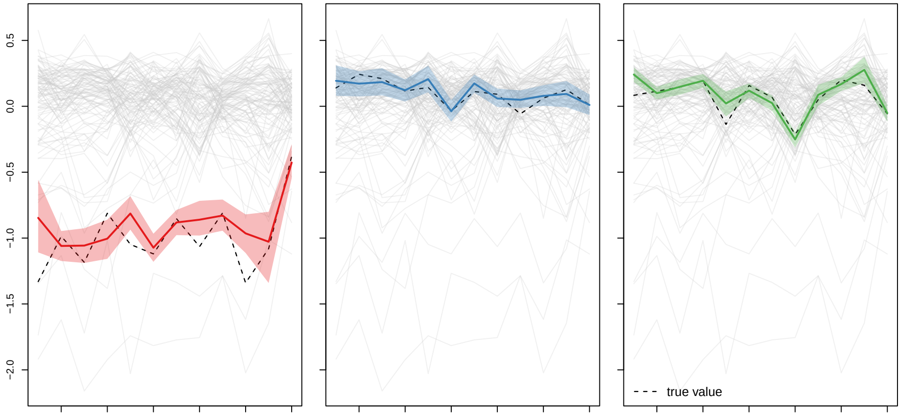
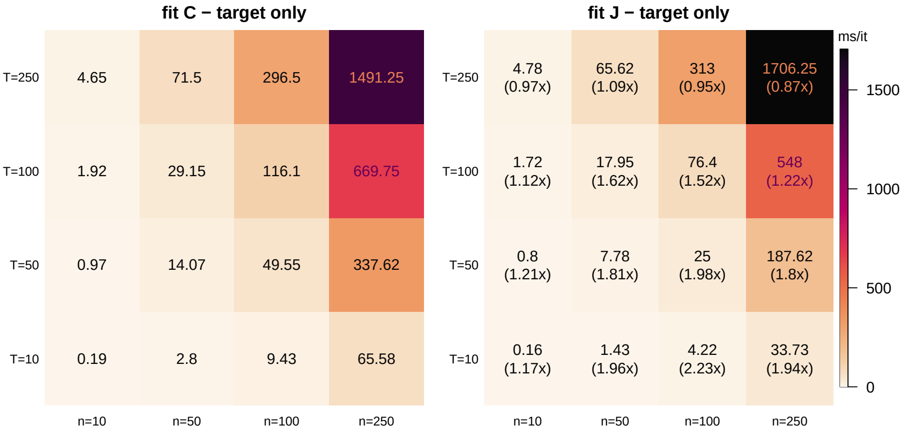
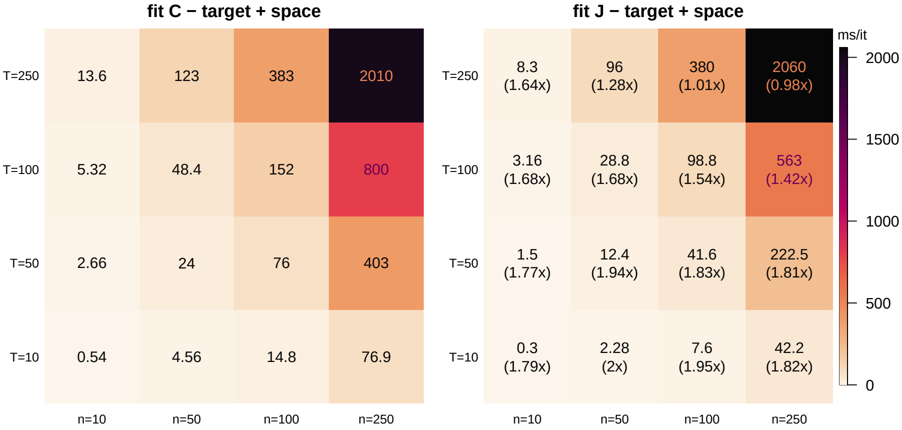


Figure: JDRPM spatially-informed fit with covariates in the likelihood and in the prior.

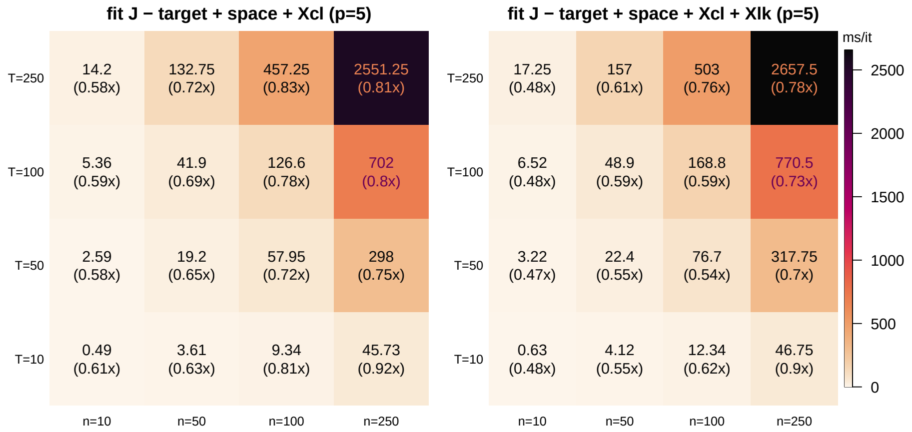
Performances in the simulated data scenario



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}