Politecnico di Milano
MSc in Mathematical Engineering
Applied Statistics

Final Project of:
Bayesian Statistics

Joint Species Distribution Models
Gibbs sampling implementation in `RCpp`

Students:
Contini Matteo
Fiorello Lorenzo
Pasquale Angelo

Advisers:
Poggiato Giovanni[a]
Dr. Corradin Riccardo

Course Lecturer:
Prof. Guglielmi Alessandra

[a]Laboratoire d'Ecologie Alpine - Inria Grenoble

Milan - February 19, 2020

Abstract

Species distribution models are used to evaluate the variables that cause the distribution and abundance of species and to predict biodiversity. Historically, such models have been fitted to each species independently. While independent models can provide useful information regarding distribution and abundance, they ignore the fact that, after accounting for environmental covariates, residual interspecies dependence persists. With stacking of individual models, misleading behaviors, may arise. In particular, individual models often imply too many species per location. Recently developed joint species distribution models have application to presence–absence, continuous or discrete abundance, abundance with large numbers of zeros, and discrete, ordinal, and compositional data. Dirichlet process is employed in these models for clustering to reduce dimension in the joint covariance structure. This last step makes computation tractable. In this project we focused on presence-absence data for the set of species recorded at each plot, we implemented a Gibbs sampler algorithm in `Rcpp` to simulate from the posteriors, we performed a posterior simulation and analyzed the results.

Contents

1 Review and comparison of Joint Species Distribution Models (JSDM)

For a quick analysis of Joint Species Distribution Models, we have focused on three main models which are actually extensions of the generalised linear modelling (GLM) framework.

1.1 Models description

- Subscript notation:
  - sites: $i = 1 \ldots, N$;
  - species: $j = 1 \ldots, S$;
  - covariates: $k = 1, ..., K$.
- Response variable $\boldsymbol{y} \in \{0, 1\}^{N \times S}$:

$$y_{ij} = \begin{cases} 1 & \text{if species } j \text{ is present at site } i \\ 0 & \text{otherwise} \end{cases}$$

- Latent variable $\boldsymbol{z} \sim \mathcal{N}_{N,S}(\mathbf{mean}, \mathbf{var})$ (associated to $\boldsymbol{y}$).
- $\boldsymbol{X} \in \mathbb{R}^{N \times K}$ matrix of measured covariates.

1.2 The core model (CM)

One of the first JSDMs was proposed by Pollock et al. (2014), and we will refer to it as the core model (CM). This model is built on the multivariate probit regression model (Chib and Greenberg (1998)), by using a latent variable parametrisation of a probit model rather than the probit link directly.

The community (i.e. the set of species) present at site i is thus characterized by the multidimensional latent variable $z_{i,\cdot}$:

$$\begin{aligned} y_{ij} &= 1(z_{ij} > 0) \\ z_{ij} &= B_{\cdot j} X_{i\cdot} + e_{ij} \\ B_{kj} &\overset{\text{ind}}{\sim} \mathcal{N}(\omega_j, \sigma_j) \\ e_{i\cdot} &\overset{\text{iid}}{\sim} \mathcal{MVN}(\mathbf{0}, \boldsymbol{R}) \end{aligned} \tag{CM}$$

where

$$\begin{cases} \omega_j \overset{\text{iid}}{\sim} \mathcal{N}(0, 100) \\ \sigma_j \overset{\text{iid}}{\sim} \mathcal{U}(0, 100) \\ \boldsymbol{R} \sim \mathcal{IW}(J + 1, \boldsymbol{I}) \quad \text{(correlation coefficients prior)} \end{cases}$$

1.3 Hierarchical Model of Species Communities (HMSC)

Hierarchical Model of Species Communities (HMSC) is a model appeared in a sequence of papers (Ovaskainen et al. (2017), Tikhonov and Ovaskainen (2017), Tikhonov, Abrego, et al. (2017)) that aim to give a very complete framework that takes into account all possible information about species in one single hierarchical model.

HMSC is a very similar to (CM), but allows the regression coefficients to be correlated:

$$
\begin{aligned}
y_{ij} &= 1(z_{ij} > 0) \\
z_{ij} &= B_{.j} X_{i.} + e_{ij} \\
B_{.j} &\overset{\text{ind}}{\sim} \mathcal{MVN}(\boldsymbol{\omega}, \boldsymbol{\Lambda}) \\
e_{ij} &= \nu_{ij} + \varepsilon_{ij} \\
\nu_{ij} &= \eta_{i.} \, \boldsymbol{A_{j.}} \\
\varepsilon_{ij} &\overset{\text{iid}}{\sim} \mathcal{MVN}(0, 1) \\
\eta_{i.} &\overset{\text{iid}}{\sim} \mathcal{MVN}(0, \boldsymbol{I}_{n_f})
\end{aligned}
\tag{HMSC}
$$

where $\boldsymbol{\omega} = (\omega_k)_{k=1,\dots,K}$ and

$$
\begin{cases}
\omega_k \overset{\text{iid}}{\sim} \mathcal{N}(0, 100) & \\
\boldsymbol{\Lambda} \sim \mathcal{IW}(5, \boldsymbol{I}) & \text{(variance-covariance matrix of the regression coefficients)} \\
\boldsymbol{R} \sim \mathcal{IW}(J+1, \boldsymbol{I}) & \text{(correlation coefficients prior)}
\end{cases}
$$

(HMSC) biggest improvement concerns the different representation of the error e_{ij}. In the (CM), the full rank matrix $\boldsymbol{R}$ represents the variation in species occurrences and co-occurrences that cannot be attributed to the responses of the species to the measured covariates.

With S species, each covariance matrix $\boldsymbol{R}$ is bijective mapping to a space of $\frac{J(J+1)}{2}$ unrestricted parameters (Lewandowski, Kurowicka, and Joe (2009)), making their estimation numerically challenging. In order to reduce the parameter space, (HMSC) uses latent factors and latent loadings. Under the classic assumption made in factor iid models that the latent factors marginally follow multivariate normal distribution $\eta_{i.} \overset{\text{iid}}{\sim} \mathcal{MVN}(0, \boldsymbol{I}_{n_f})$, the latent loadings provide then a parametrisation of $\boldsymbol{R}$ as $\boldsymbol{R} = \boldsymbol{A}^T \boldsymbol{A} + \boldsymbol{I}_S$, where $\boldsymbol{A} = \{A_{ij}\}$ is the $S \times n_f$ matrix containing all latent loadings. The utility of the latent factor approach comes from the dimension-reduced parametrization of $\boldsymbol{R}$ in case where $n_f \ll S$. Instead of fixing the number of latent factors n_f, (HMSC) treats n_f as an unknown parameter through the shrinkage approach of Bhattacharya, Pati, and Dunson (2014). This variance decomposition could be considered similar to a linear regression where the latent loadings $A_{j,q}$ are the parameters of the regression, and the latent factors are interpreted to model some missing covariates, which have an impact on the species occurrences and are not represented in the matrix.

1.4 Generalized Joint Additive Model (GJAM)

GJAM is a Joint Species Distribution Model that aims to fit all type of response data, using a latent variable. This is an important feature: since in ecology the collection of data can be very heterogeneous, it is suitable to have a single model to deal with multifarious data. For presence-absence data it is a multivariate probit regression model that takes on two different forms depending on S , the number of species to be modeled, for the same reasons we discussed above: when the number of species S is big, the model suffers from the "curse of dimensionality". The small dataset form (i.e. when S is small) is equivalent to the core model (CM), but the regression coefficients B_{jk} are indipendent and vague:

$$\begin{aligned} y_{ij} &= 1(z_{ij} > 0) \\ z_{ij} &= B_{.j} X_{i.} + e_{ij} \\ B_{.j} &\overset{\text{ind}}{\sim} \mathcal{MVN}(\mathbf{0}, 100\, \boldsymbol{I}) \\ e_{i.} &\overset{\text{iid}}{\sim} \mathcal{MVN}(\mathbf{0}, \boldsymbol{R}) \\ \boldsymbol{R} &\sim \mathcal{IW}(J+1, \boldsymbol{I}) \end{aligned} \tag{GJAM1}$$

The big dataset form (i.e. when S is big and dimension reduction is needed) proposes a latent factor approach similar to (HMSC):

$$\begin{aligned} y_{ij} &= 1(z_{ij} > 0) \\ z_{ij} &= B_{.j} X_{i.} + \boldsymbol{A} \\ V_{ij} &= \boldsymbol{X_{i.}}\, \boldsymbol{B_{j.}}^T + e_{ij} \end{aligned} \tag{GJAM2}$$

2 GJAM using Bayesian nonparametric priors

Taylor-Rodríguez et al. (2017) made use of the clustering property of the **Dirichlet Process (DP)** to allow some rows of $\boldsymbol{A}$ to be common, which corresponds to clustering species in their dependence behavior.

We briefly recall the well known construction of a DP through the stick-breaking procedure.

Consider two independent families $\{V_k\}$ and $\{Z_k\}$ of random variables:

$$V_k \overset{iid}{\sim} Be(1, \alpha) \qquad Z_k \overset{iid}{\sim} H \qquad k = 1, 2, \ldots. \tag{1}$$

We can now define the random weights as:

$$\begin{aligned} p_1 &= V_1, \\ p_k &= V_k \prod_{j=1}^{k-1} (1 - V_j), \qquad k = 2, \ldots \end{aligned} \tag{2}$$

The construction of p can be understood metaphorically as follows. Starting with a stick of length 1, we break it at V_1, assigning p_1 to be the length of stick we just broke

off. Now we recursively break the other portion proportionally to V_2, V_3, to obtain p_2, p_3 and so forth. Notice that $\sum_{k=1}^{\infty} p_k = 1$ *a.s.*.
Then the stick-breaking representation of the Dirichlet process can be written as:

$$G := \sum_{k=1}^{\infty} p_k \delta_{Z_k}. \tag{3}$$

The infinite dimension of the stick-breaking representation comes with a cost: we cannot directly sample from it. To be simulated, the Dirichlet process is truncated at a level N in such a way:

$$G := \sum_{k=1}^{N} p_k \delta_{Z_k}. \tag{4}$$

Based on employing the truncated stick-breaking, it yields the GJAM complete model:

$$\begin{aligned}
\boldsymbol{V}_i \,|\, \boldsymbol{k}, \boldsymbol{Z}, \boldsymbol{w}_i, \boldsymbol{B}, \sigma_\epsilon^2 &\sim \mathcal{N}_S(\boldsymbol{B}_i + \boldsymbol{Q}(k)\, \boldsymbol{Z}\, \boldsymbol{w}_i, \sigma_\epsilon^2\, \boldsymbol{I}_S), \quad \text{for } i = 1, \ldots, n \\
[\boldsymbol{B}, \sigma_\epsilon^2] &\propto \frac{1}{\sigma_\epsilon^2} \\
\boldsymbol{w}_i &\sim \mathcal{N}_r(\boldsymbol{0}, \boldsymbol{I}_r) \\
k_l | \mathbf{p} &\overset{\text{iid}}{\sim} \sum_{j=1}^{N} p_j \delta_j(k_l), \quad \text{for } l = 1, \ldots S \\
\boldsymbol{Z}_j \,|\, \boldsymbol{D}_z &\overset{\text{iid}}{\sim} \mathcal{N}_r(\boldsymbol{0}, \boldsymbol{D_z}), \quad \text{for } j = 1, \ldots N \\
\boldsymbol{p} &\sim \mathcal{GD}_N(a_\alpha, b_\alpha) \\
\boldsymbol{D_z} &\sim \mathcal{IW}(2 + r - 1, 4\text{diag}(\frac{1}{\eta_1}, \ldots, \frac{1}{\eta_r})) \\
\eta_h &\sim \mathcal{IG}(\frac{1}{2}, \frac{1}{10^4}), \quad \text{for } h = 1, \ldots, r
\end{aligned} \tag{5}$$

where $\mathcal{GD}_N(a_\alpha, b_\alpha)$ corresponds to the N-dimensional Generalized Dirichlet process with $a_\alpha = (\frac{\alpha}{N}, \ldots, \frac{\alpha}{N})$ and $b_\alpha = (\frac{\alpha(N-1)}{N}, \frac{\alpha(N-2)}{N}, \ldots, \frac{\alpha}{N})$.

2.1 Adaptation to binary responses (Presence/Absence)

To work with binary responses, we resort to the adaptation proposed in the paper by Taylor and Rodriguez. This modification is known in the literature as the parameter-expansion data-augmentation (PX-DA) algorithm (Liu and Wu, 1999; Lawrence et al., 2008; Schliep and Hoeting, 2015). It allows us to use the machinery proposed for the continuous case as a latent model. The PX-DA strategy was also considered by (Clark et al., 2016) to model non-continuous responses. In either case, approximation is needed to handle a large collection of species. The augmentation consists of introducing multivariate normal latent random variables, which are used to obtain a full conditional posterior

density where the entire covariance matrix can be sampled. The sampled covariance is then re-scaled as a correlation matrix to accommodate the identifiability constraints imposed by the probit link. In our case, the approach is further modified to accommodate the dimension reduction step as described below. Again, let $\boldsymbol{\Sigma}^\star = \boldsymbol{A}\,\boldsymbol{A}' + \sigma^2_\epsilon\,\boldsymbol{I}$ for some $S \times r$ matrix $\boldsymbol{A}$, and denote by $\boldsymbol{R} = \boldsymbol{D}^{-\frac{1}{2}}\,\boldsymbol{\Sigma}^\star\,\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the diagonal matrix containing $\text{diag}(\boldsymbol{\Sigma}^\star)$. We can use data augmentation with the binary likelihood, assuming that, for plot i, $\boldsymbol{V} \sim \mathcal{N}_{n\times S}(\boldsymbol{X}\,\boldsymbol{B}', \boldsymbol{R}, \boldsymbol{I}_n)$, where $\boldsymbol{X}$ is the $n \times p$ matrix of predictors and $\boldsymbol{B}$ is the $S \times p$ matrix of regression coefficients. Assume that the matrix of binary responses is given by $\boldsymbol{Y} = [\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n]$, where $\boldsymbol{Y}_i = (Y_{i1}, \ldots, Y_{iS})$ for $i = 1, \ldots, n$. Recall that we connect $\boldsymbol{Y}_i$ with $\boldsymbol{V}_i$ through $Y_{ij} = I(V_{ij} > 0)$ so that the contribution to the likelihood for species j in plot i is $I^{Y_{ij}}_{\{V_{ij}>0\}} I^{1-Y_{ij}}_{\{V_{ij}\le 0\}}$. With $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iS})$ the binary vector of observed presences indicators at plot i, we have

$$\Pr(\boldsymbol{Y}_i = \boldsymbol{y}_i) = \int_{\Gamma(y_{iS})} \cdots \int_{\Gamma(y_{i1})} (2\pi)^{-\frac{S}{2}}\,|\boldsymbol{R}|^{-\frac{1}{2}} \times \exp\big\{-\frac{1}{2}(\boldsymbol{V}_i - \boldsymbol{B}\,x_i)'\,\boldsymbol{R}^{-1}(\boldsymbol{V}_i - \boldsymbol{B}\,x_i)\big\}\mathrm{d}\boldsymbol{V}_i \quad (6)$$

where $\boldsymbol{V}_i$ is the ith row of $\boldsymbol{V}$, and $\Gamma(y_{ij})$ is $(-\infty, 0)$ if $y_{ij} = 0$ and $(0, \infty)$ if $y_{ij} = 1$. Now, let $\boldsymbol{V}^\star_i = \boldsymbol{D}^{\frac{1}{2}}\,\boldsymbol{V}_i$ and note that $\boldsymbol{V}^\star_i \sim \mathcal{N}_S(\boldsymbol{B}\,x_i, \boldsymbol{\Sigma}^\star)$, where $\boldsymbol{B}^\star = \boldsymbol{D}^{\frac{1}{2}}\,\boldsymbol{B}$. This change of variable doesn't affect the probabilities for $\boldsymbol{Y}_i$. Hence,

$$\Pr(\boldsymbol{Y}_i = \boldsymbol{y}_i) = \int_{\Gamma(y_{iS})} \cdots \int_{\Gamma(y_{i1})} (2\pi)^{-\frac{S}{2}}\,\left|\boldsymbol{\Sigma}^\star\right|^{-\frac{1}{2}} \times \exp\big\{-\frac{1}{2}(\boldsymbol{V}^\star_i - \boldsymbol{B}^\star\,x_i)'\boldsymbol{\Sigma}^{\star -1}(\boldsymbol{V}^\star_i - \boldsymbol{B}^\star\,x_i)\big\}\mathrm{d}\boldsymbol{V}^\star_i \quad (7)$$

which in turn implies the expanded likelihood given by

$$\begin{aligned}\mathcal{L}(\boldsymbol{B}^\star, \boldsymbol{\Sigma}^\star, \boldsymbol{V}^\star \,|\, \boldsymbol{Y}) = |\boldsymbol{\Sigma}|^{-\frac{n}{2}} \Bigg(\prod_{i=1}^{n} \exp\{-\frac{1}{2}(\boldsymbol{V}^\star_i - \boldsymbol{B}^\star\,x_i)'\boldsymbol{\Sigma}^{\star -1}(\boldsymbol{V}^\star_i - \boldsymbol{B}^\star\,x_i)\} \\ \times \prod_{j=1}^{S} I^{y_{ij}}_{\{V^\star_{ij}>0\}} I^{1-y_{ij}}_{\{V^\star_{ij}\le 0\}}\Bigg)\,[\boldsymbol{B}^\star]\,[\boldsymbol{\Sigma}^\star]\end{aligned} \quad (8)$$

As in the continuous response case, we can introduce an $n \times r$ matrix of standard normal random variables $\boldsymbol{W}$, such that $\boldsymbol{V}^\star \,|\, \boldsymbol{B}^\star, \boldsymbol{A}, \boldsymbol{W}, \sigma^2_\epsilon \sim \mathcal{N}_{n\times S}(\boldsymbol{X}\,\boldsymbol{B}^{\star\prime} + \boldsymbol{W}\,\boldsymbol{A}', \sigma^2_\epsilon\,\boldsymbol{I}_S, \boldsymbol{I}_n)$, where $\mathcal{N}_{a\times b}(\mathbf{M}, \boldsymbol{V}_{\text{col}}, \boldsymbol{V}_{\text{row}})$ represents the $a \times b$ -dimensional matrix normal distribution with mean $\mathbf{M}$, and column and row covariance matrix $\boldsymbol{V}_{\text{col}}$ and $\boldsymbol{V}_{\text{row}}$, respectively. Hence, the expanded likelihood can now be expressed as

$$\begin{aligned}&\mathcal{L}_{\mathbf{PA}}(\boldsymbol{B}^\star, \boldsymbol{W}, \boldsymbol{A}, \sigma^2_\epsilon, \boldsymbol{V}^\star \,|\, \boldsymbol{Y}) \\ &\quad\propto (\sigma^2_\epsilon)^{-\frac{nS}{2}} \prod_{i=1}^{n} \exp\{-\frac{1}{2\sigma^2_\epsilon}||\,\boldsymbol{V}^\star_i - \boldsymbol{B}^\star\,\mathbf{x}_i - \boldsymbol{A}\,\boldsymbol{w}_i\,||^2 - \frac{1}{2}||\,\boldsymbol{w}_i\,||^2\} \times \ldots \\ &\qquad\qquad \times \prod_{j=1}^{S} I^{y_{ij}}_{\{V^\star_{ij}>0\}} I^{1-y_{ij}}_{\{V^\star_{ij}\le 0\}}\,[\boldsymbol{B}^*]\,[\boldsymbol{A}]\,\left[\sigma^2_\epsilon\right]\end{aligned} \quad (9)$$

With this expanded likelihood, the sampling algorithm becomes

1. Sample $\boldsymbol{V}_i^\star \sim tr\,\mathcal{N}_S(\boldsymbol{B}^\star\,\mathbf{x}_i + \boldsymbol{A}\,\boldsymbol{w}_i, \sigma_\epsilon^2\,\boldsymbol{I}_S; \Gamma(y_i))$. This proposal density is convenient as it corresponds to drawing from univariate truncated normal random variables.

2. Draw $\boldsymbol{A} = \boldsymbol{Q}(k)\,\boldsymbol{Z}$ and σ_ϵ^2 as in the continuous case, where the full conditional densities for $\boldsymbol{A}$ and σ_ϵ^2 depend on $\boldsymbol{V}^\star$, $\boldsymbol{B}^\star$, and $\boldsymbol{W}$.

3. Differently from Taylor and Rodriguez, we assume a gaussian prior on $\boldsymbol{B}^\star$, and with some calculations we get this full conditional posterior distribution $\beta_j^\star \sim N\left(\left(\frac{1}{\sigma^2}I + \frac{1}{\sigma_\epsilon^2}X'X\right)^{-1}\frac{1}{\sigma_\epsilon^2}X'\left(V_j^\star - WA_j\right), \left(\frac{1}{\sigma^2}I + \frac{1}{\sigma_\epsilon^2}X'X\right)^{-1}\right)$ for each species $j = 1, \ldots, S$

4. Finally, obtain the variables on the correlation scale using the transformations $\boldsymbol{V} = \boldsymbol{D}^{-\frac{1}{2}}\,\boldsymbol{V}^\star$, $\boldsymbol{B} = \boldsymbol{D}^{-\frac{1}{2}}\,\boldsymbol{B}^\star$ and $\boldsymbol{R} = \boldsymbol{D}^{-\frac{1}{2}}(\boldsymbol{A}\,\boldsymbol{A}' + \sigma_\epsilon^2\,\boldsymbol{I})\,\boldsymbol{D}^{-\frac{1}{2}}$.
$\boldsymbol{V} = \boldsymbol{D}^{-\frac{1}{2}}\,\boldsymbol{V}^\star \rightarrow \boldsymbol{V} = \boldsymbol{V}^\star\,\boldsymbol{D}^{-\frac{1}{2}}$

3 Sampling strategy

In order to sample from the posterior distributions we decided to build a Gibbs sampler. To do so, we needed all full conditionals. Our main function calls 8 different "STEP" functions in the loop, each one of them sampling one full conditional.

3.1 Full Conditionals

- $[\boldsymbol{Z} \mid \boldsymbol{D_z}, \boldsymbol{B}, \boldsymbol{W}, \sigma_\varepsilon^2, \boldsymbol{V}]$: The posterior for each row of $\boldsymbol{Z}$ depends on whether or not the row considered was chosen to be at least one row from A. That is, for $j = 1, \ldots N$

  - If $j \notin k$, sample
  $$\boldsymbol{Z}_j \sim \mathcal{N}_r(\mathbf{0}, \boldsymbol{D_z}) \tag{1.1}$$
  - If $j \in k$, let $S_j = \{l = 1, \ldots S \text{ s.t. } k_l = j\}$ and let
  $$\boldsymbol{Z} \mid \boldsymbol{D_z}, \boldsymbol{B}, \boldsymbol{W}, \sigma_\varepsilon^2, \boldsymbol{V} \overset{\text{ind}}{\sim} \mathcal{N}_r(\mu_{z_l}, \Sigma_{\boldsymbol{Z}_j}) \tag{1.2}$$
  where $\Sigma_{\boldsymbol{Z}_j} = \left(\frac{|S_j|}{\sigma_\epsilon^2}\,\boldsymbol{W}^T\,\boldsymbol{W} + \boldsymbol{D_z}^{-1}\right)^{-1}$, $\mu_{\boldsymbol{Z}_j} = \Sigma_{\boldsymbol{Z}_j}\,\boldsymbol{W}^T\,\frac{1}{\sigma_\varepsilon^2}\sum_{l \in S_j}(\boldsymbol{V}^{(l)} - \boldsymbol{X}\,\boldsymbol{\beta}_l)$, and finally, $\boldsymbol{V}^{(l)}$ and $\boldsymbol{\beta}_l$ are the l-th column of the matrix $\boldsymbol{V}$ and the l-th row of $\boldsymbol{B}$, respectively.

- $[\boldsymbol{W} \mid \boldsymbol{B}, \boldsymbol{A}, \sigma_\varepsilon^2, \boldsymbol{V}]$
$$\boldsymbol{w}_i \mid \boldsymbol{B}, \boldsymbol{A}, \sigma_\epsilon^2, \boldsymbol{V}_i \sim \mathcal{N}_r\left(\Sigma_{\boldsymbol{W}}\,\boldsymbol{A}\,\frac{1}{\sigma_\varepsilon^2}(\boldsymbol{V}_i - \boldsymbol{B}\,x_i), \Sigma_{\boldsymbol{W}}\right) \tag{2}$$
where $\Sigma_{\boldsymbol{W}} = (\frac{1}{\sigma_\epsilon^2}\,\boldsymbol{A}^T\,\boldsymbol{A} + I_r)^{-1}$

- $[\boldsymbol{k} \,|\, \boldsymbol{p}, \boldsymbol{B}, \boldsymbol{Z}, \sigma^2_\varepsilon, \boldsymbol{V}]$

$$[\boldsymbol{k} \,|p, \boldsymbol{B}, \boldsymbol{Z}, \sigma^2_\epsilon, \boldsymbol{V}] = \prod_{l=1}^{q} \left\{ \sum_{j=1}^{N} p_{lj} \delta_j(k_l) \right\} \tag{3}$$

with $p_{lj} \propto p_j \times \exp[-\frac{1}{2\sigma^2_\epsilon} || \, \boldsymbol{V}^{(l)} - \boldsymbol{X} \, \beta_l - \boldsymbol{W} \, Z_j ||^2]$

- $[\boldsymbol{p} \,|\, \boldsymbol{k}]$ The full conditional posterior for $\boldsymbol{p}$, given conjugancy of the $\mathcal{GD}$ distribution with multinomial sampling, the draws of $\boldsymbol{p}$ are

$$\begin{aligned} p_1 &= \xi_1 \\ p_j &= (1-\xi_1) \dots (1-\xi_{j-1}) \, \xi_j \;\text{ for }\; j = 2, 3, \dots, N-1 \\ p_N &= 1 - \sum_{j=1}^{N-1} p_j \end{aligned} \tag{4}$$

with

$$\xi_j \overset{\text{ind}}{\sim} Beta \left(\frac{\alpha}{N} + \sum_{l=1}^{S} I_{(k_l=j)}, \frac{N-j}{N}\alpha + \sum_{s=j+1}^{N} \sum_{l=1}^{S} I_{(k_l=s)} \right) \quad \text{for } j = 1, \dots, N-1$$

- $[\sigma^2_\varepsilon | \, \boldsymbol{A}, \boldsymbol{W}, \boldsymbol{B}, \boldsymbol{V}]$

$$\sigma^2_\epsilon | \, \boldsymbol{A}, \boldsymbol{W}, \boldsymbol{B}, \boldsymbol{V} \sim \mathcal{IG} \left(\frac{nS+\nu}{2} + 1, \frac{\sum_{i=1}^{n} || \, \boldsymbol{V}_i - \boldsymbol{B} \, x_i - \boldsymbol{A} \, w_i ||^2}{2} + \frac{\nu}{G^2} \right) \tag{5}$$

- $[\boldsymbol{D_z} \,|\, \boldsymbol{Z}]$: By conjugacy of the prior $\boldsymbol{D_z}$ with the normal prior for $\boldsymbol{Z}$, the full conditional for $\boldsymbol{D_z}$ is

$$\boldsymbol{D_z} \,|\, \boldsymbol{Z} \sim \mathcal{IW} \left(\boldsymbol{D_z} \,|2 + r + N - 1, \boldsymbol{Z}' \, \boldsymbol{Z} + 4\text{diag} \left\{ \frac{1}{\eta_1}, \dots, \frac{1}{\eta_r} \right\} \right) \tag{6}$$

- $[\boldsymbol{V_{i.}} \,|\, \boldsymbol{Y_{i.}}, \boldsymbol{B}, \boldsymbol{R}]$

$$\boldsymbol{V_{i.}} \sim \mathcal{N}_S(\boldsymbol{B} \, x_i + \boldsymbol{A} \, w_i, \sigma^2_\epsilon I_S; \Gamma(y_i)) \tag{7}$$

- $[\boldsymbol{B} \,|\, \boldsymbol{X}, \boldsymbol{V}]$

Sample the vector of regression coefficients $\boldsymbol{B}_j$ for each species j from a multivariate normal distribution:

$$\boldsymbol{B}_j \sim \mathcal{N}((\sigma \, \boldsymbol{I} + \boldsymbol{X_j}' \, \boldsymbol{X_j})^{-1} \, \boldsymbol{X_j}' \, \boldsymbol{V_{.j}}, (\sigma \, \boldsymbol{I} + \boldsymbol{X_j}' \, \boldsymbol{X_j})^{-1}) \tag{8}$$

In this notation we write $\boldsymbol{X_j}$ because we suppose that every species can depend on different covariates, so will exist a different design matrix $\boldsymbol{X_j}$ for every species j. In our case we will drop the index j and consider a unique design matrix $\boldsymbol{X}$

3.2 Additional steps 7 and 8: priors and prosteriors

Differently from Taylor and Rodriguez, here a proper prior has been choosen for $\boldsymbol{B}$.

$$[\boldsymbol{B}, \sigma_\epsilon^2] = \pi[\boldsymbol{B}] \times \frac{1}{\sigma_\epsilon^2} \tag{10}$$

where $\pi[\boldsymbol{B}]$ is such that all the coefficients are independent across species:

$$\forall j = 1, \ldots, S \qquad \boldsymbol{\beta_j} \overset{iid}{\sim} N(\boldsymbol{0}, \sigma^2 \, \boldsymbol{I}) \tag{11}$$

where $\boldsymbol{\beta_j}$ is the j-th row of $\boldsymbol{B}$ (if $\boldsymbol{B}$ is $S \times k$). σ is an hyperparameter that gives the degree of uninformativeness of the prior. You can take $\sigma = 10$.

- $[\boldsymbol{V}^\star \,|\, \boldsymbol{B}^\star, \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{W}, \boldsymbol{Y}]$: for each site $i = 1, \ldots, n$ and each species $j = 1, \ldots, S$ we sample $V_{i,j}^\star$ from a univariate truncated normal:

  $$\boldsymbol{V}_{i,j}^\star \,|\, \boldsymbol{B}^\star, \boldsymbol{X}, \boldsymbol{A}, \boldsymbol{W}, y_{i,j} \sim trunc.N(\boldsymbol{B_{j.}}^\star \, \boldsymbol{x_i} + \boldsymbol{A} \, \boldsymbol{w_i}, \sigma_\epsilon^2) \tag{12}$$

  Where the normal is truncated to the positive axis if $y_{i,j} = 1$ and to the negative axis if $y_{i,j} = 0$.

- $[\boldsymbol{B}^\star \,|\, \boldsymbol{V}^\star, \boldsymbol{X}, \boldsymbol{A}, ...]$: for each species $j = 1, \ldots, S$ we sample $\beta_j^\star$ from a multivariate normal

  $$\boldsymbol{\beta_j}^\star \sim N((\frac{1}{\sigma^2} \, \boldsymbol{I} + \frac{1}{\sigma_\epsilon^2} \, \boldsymbol{X}' \, \boldsymbol{X})^{-1} \frac{1}{\sigma_\epsilon^2} \, \boldsymbol{X}'(\boldsymbol{V_{.j}}^\star - \boldsymbol{W} \, \boldsymbol{A_{j.}}), (\frac{1}{\sigma^2} \, \boldsymbol{I} + \frac{1}{\sigma_\epsilon^2} \, \boldsymbol{X}' \, \boldsymbol{X})^{-1}) \tag{13}$$

  Where $\boldsymbol{W}$ is the $n \times r$ matrix whose lines are $\boldsymbol{w_i}$.
  Notice that this is a mix between equation (3.A.3) by Golding and step 3 in section 3.2 by Taylor-Rodriguez.

- $[\sigma_\epsilon^2 | rest]$ as in Taylor and Rodriguez appendix

- Finally, obtain the variables on the correlation scale using the following transformations.
  Let $\boldsymbol{D}$ be the diagonal matrix $\boldsymbol{D} = \text{diag}(\boldsymbol{\Sigma}^\star)$, with $\boldsymbol{\Sigma}^\star = \boldsymbol{A} \, \boldsymbol{A}' + \sigma_\epsilon^2$.
  Then, $\boldsymbol{V} = \boldsymbol{D}^{-1/2} \, \boldsymbol{V}^\star$, $\boldsymbol{B} = \boldsymbol{D}^{-1/2} \, \boldsymbol{B}^\star$ and $\boldsymbol{R} = \boldsymbol{D}^{-1/2}(\boldsymbol{A} \, \boldsymbol{A}' + \sigma_\epsilon^2 \, \boldsymbol{I}) \, \boldsymbol{D}^{-1/2}$.
  This step is needed to the identifiablily issue of the probit link, where the variance covariance matrix has to be a correlation matrix.

3.3 Pseudo-code

First of all the pseudo-code of the Gibbs sampler we aim at implementing reads:

function GJAM_GIBBS_SAMPLER( )
  for $j = 1, \ldots, N$ **do**
   resample Z_j according to the following
    if $j \notin \boldsymbol{k}$ **then** sample from 1.1
    else sample Z_j from 1.2
    end if
  end for
  for $i = 1, \ldots, N$ **do** resample $\boldsymbol{w_i}$ from 2
  end for
   Resample the vector of labels $\boldsymbol{k}$ from 3
   Resample the vector of labels $\boldsymbol{p}$ from 4, thus
  for $j = 1, \ldots, N-1$ **do** sample ξ_j and thus p_j
  end for
   $p_N = 1 - \sum_{j=1}^{N-1} p_j$
   sample σ_ϵ^2 from 5
   sample $\boldsymbol{D_z}$ from 6
   sample $\boldsymbol{V}$ from 7
   sample $\boldsymbol{B}$ from 8
end function

4 Implementation in `Rcpp`

The chosen programming environment is `R` and, with the aim of optimizing computational efficiency, the integration with a `C++` code through the `Rcpp` package has been made.

Our code is structured as follows. One main `R` function called `gjam_gibbs_sampler` is in charge of the sampler. Each step analyzed in 3.1 is performed in separate `Rcpp` functions in order to control each posterior draw at the best.

`fit_gjam_gibbs ← function(alpha0,ndraws,burnin,N_stick,r,S,n_sites,x,Y)`

- `alpha0` is the mass parameter α for the Dirichlet process.
- `ndraws, burnin` the iteration parameters, the Gibbs sampler will run for a total of `ndraws` iterations and the first `burnin` will be discarded.
- `N_stick` is the level of truncation of the Dirichlet process.
- `r` is the number of latent factors considered.
- `S` is the number of species.
- `n_sites` is the number sites.
- `x` is the `n_sites`$\times 2$ matrix of the covariates, in our case we used the intercept and one covariate. `Y` is the `n_sites`$\times S$ absence/presence matrix for species. Both these two matrices are generated in our code through a simulation function.

Our benchmark and reference code is the one developed by Clark et al. (2017) in the R library gjam. Since some functionalities of our Gibbs are based on this library, as well as other well-known packages, they are required for our code to work. The user will find an automatic check and installation of the required packages:

```
needed_packages <- c("MASS","coda","ggmcmc","extrafont","mgcv","
   mvtnorm", "matlib", "devtools", "MCMCpack", "gjam", "invgamma",
   "MixMatrix", "tictoc", "corpcor")

new_packages <- needed_packages[!(needed_packages %in%
   installed.packages()[, "Package"])]
if (length(new_packages)) install.packages(new_packages)
lapply(needed_packages, require, character.only = TRUE)}
```

Code 1: Installation of required packages

At first, the algorithm has been written completely in R. Then, we have focused on implementing it in C++. Even in steps where only few operations where needed we realized that simple operations like calculate the inverse of a matrix

```
  sigmaBetaj = solve(1/(sigmaB^2) * diag(n_cov) + 1/sigmaeps2 * t(x
     ) %*% x)
```

Code 2: Excerpt from Step8.R

or simulate from a normal distribution

```
  B_star[j,] <- rmvnorm ( n = 1, mean = muBetaj, sigma = sigmaBetaj
      )
```

Code 3: Another excerpt from Step8.R

where strongly speeded up in their Rcpp counterpart

```
sigmaBetaj = solveRcpp(1/(sigmaB^2) * diag(n_cov) + 1/sigmaeps2 *
   t(x) %*% x)

B_star[j,] <- rmvnormRcpp ( n = 1, mu = muBetaj, sigma =
   sigmaBetaj )
```

Code 4: Excerpt from Step8Rcpp.R

where for simple but useful operations like solveRcpp we've exploited Armadillo library

```
Inverse of A
1  arma::mat solveRcpp(arma::mat A) {
2  arma::mat AA(A);
```

```
  arma::mat Ainv = arma::inv_sympd(AA);
  return Ainv;
```

Here below an example of R code and its C++ counterpart for the Step 1 of the Gibbs sampler, i.e. drawing from the full conditional of $\boldsymbol{Z}_j$, atoms in the Dirichlet Process approximation.

```
  for ( j in 1:N_stick ) {
    if (!(j %in% k)) {
      Z[j,] <- rmvnorm ( n = 1, mean = rep(0, times = r), sigma =
         Dz )
      Cardinality_S[j]<- 1
    }
    else { # if j in k
      Cardinality_S[j]<- sum(j==k)
      Sigma_Zj = solve(Cardinality_S[j]/sigmaeps2 * t(W) %*% W +
         solve(Dz) )
      mu_Zj = c(0)
      for (l in 1:S) {
        if (k[l] == j){
          mu_Zj = mu_Zj + 1/sigmaeps2 * Sigma_Zj %*% t(W) %*%
          ( t(t(V[,l])) - x %*% B[l,] )
        }
        Q[l,k[l]]<- 1
      }
      Sigma_Zj <- make.positive.definite(Sigma_Zj, tol=1e-3)
      # because of machine precision the matrix is not recognized
         as symmetric positive definite
      # we overcome this by making sure your det(Sigma_Zj) returns
         a positive value. One way is to add some variance in all
         directions.
      Sigma_Zj[lower.tri(Sigma_Zj)] = t(Sigma_Zj)[lower.tri(Sigma_
         Zj)]
      Z[j,]<-rmvnorm ( n = 1, mean = mu_Zj, sigma = Sigma_Zj )
    }
  }
  A<- Q %*% Z
  A
}
```

Code 5: R code

```
arma::mat fnZRcpp(arma::vec kk, arma::mat Yk, arma::mat Xk, arma::mat Dk,
                  arma::mat Bk, arma::mat Wk, double sigmasqk, int
                  ↪  Nz) {
```

```
  vec k = kk;
  mat Y(Yk);
  mat X(Xk);
  mat B(Bk);
  mat W(Wk);
  mat D(Dk);
  double sigmasq = sigmasqk;

  int N = Nz;
  int r = W.n_cols;
  int nn = Y.n_rows;
  //int q = Y.n_cols;
  int s = 0;

  vec kstar = unique(kk) - 1;
  vec knotstar(N - kstar.size());
  for(int j = 0; j < N; ++j){
    if(all(kstar != j)){
      knotstar[s] = j;
      s = s+1;
    }
  }

  mat Z(N,r);
  mat WtW = W.t()*W;
  mat Dinv = arma::inv_sympd(D);
  int nkk = kstar.size();

  uvec J;
  int js;
  mat CovZj(r,r);
  vec ssY(nn);
  vec meanZj(r);
  mat tempmat;
  vec tempvec(r);

  for(int i = 0; i < nkk; ++i){
    J = find(k == (kstar(i) + 1));
    js = J.size();
    CovZj = arma::inv_sympd(as_scalar(js/sigmasq)*WtW + Dinv);
    tempmat = Y.cols(J) - X*B.rows(J).t();
    if(J.size() > 1){
      ssY = arma::sum(tempmat,1);
    } else {
```

```
      ssY = tempmat;
    }

    tempvec = vectorise(as_scalar(1/sigmasq)*CovZj*W.t()*ssY);
    meanZj = tempvec;
    Z.row(kstar(i)) = rmvnormRcpp(1, meanZj, CovZj);
    //arma::repmat(meanZj, 1, 1).t() + randn(1, r) * chol(CovZj);
    J.reset();
  }

  int nss = knotstar.size();

  for(int i = 0; i < nss; ++i) {
    Z.row(knotstar(i)) = rmvnormRcpp(1, zeros<arma::vec>(r), D);
    //repmat(zeros<arma::vec>(r), 1, 1).t() + randn(1, r) * chol(D);
  }

  //rmvnormArma(knotstar.size(),zeros<vec>(r),D);

  return Z;
}
```

5 Simulations

In order to simulate our data we used the function `simulation_fun` given us by Dr.Poggiato where the input parameters are :

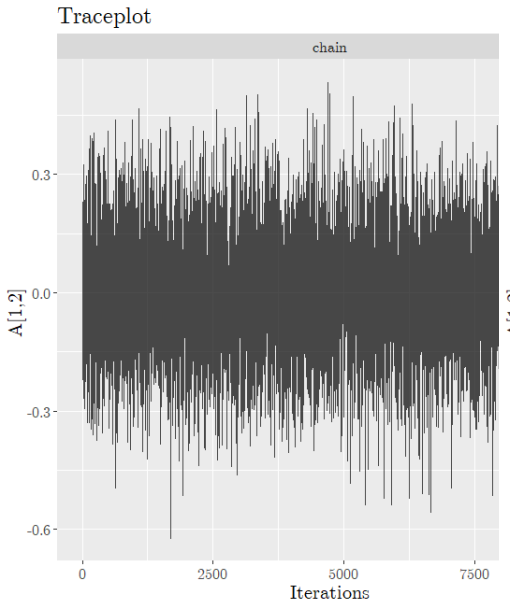
- `Sp` = number of species
- `nsamples` = number of sites
- `r` = number of latent factor
- `K_t` = number of clusters in the matrix A_{true}

```
simulation_fun<-function(Sp=S,nsamples=n_sites, r=4, K_t=4){
  S<-Sp
  n<- nsamples
  #iterations<-it

  #create the design matrix: here we use the intercept and one
     covariate (scaled!)
  X<-cbind(rep(1,n),scale(runif(0,100,n=n)))
```

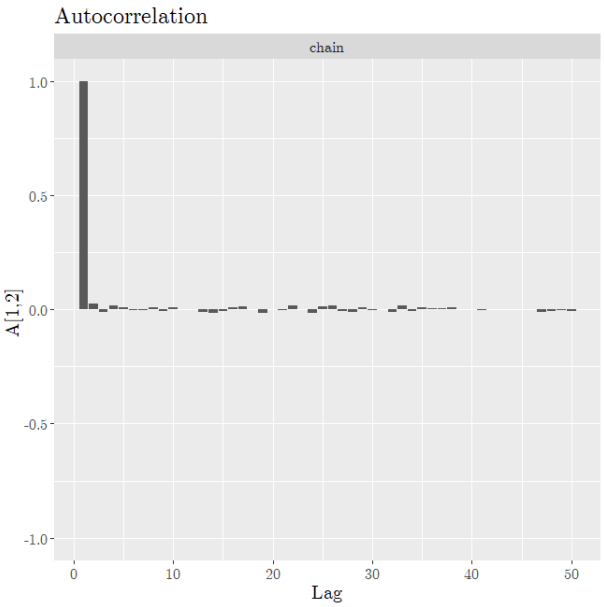
```
  #create the coefficient matrix B (Sx2)
  idx<-sample(S)
  B_0<-scale(seq(0,100,length.out=S)[idx]) #intercept
  idx<-sample(S)
  B_1<-scale(seq(0,100,length.out=S)[idx]) #covariate coefficient
  B<-cbind(B_0,B_1) #Coefficient matrix
  L<-X%*%t(B) #We create the mean by multiplying B with the design
     matrix X


  #We create A and then Sigma
  A<-matrix(NA,nrow=K_t,ncol=r) #A is the matrix with the atoms
     only. Initialization.
  sig=matrix(runif(n=r*r),ncol=r) #sig is the variance covariance
     matrix of Z.
  for(i in 1:K_t){
    A[i,]<-mvrnorm(n = 1, rep(0,r), Sigma=1*diag(r)) # We sample
       the unique values of A
  }
  idx<-sample((1:K_t),S,replace=T) #idx represents the labels
  Lambda<-A[idx,] #We expand A using the labels to obtain Lambda (
     what we used to call A)
  Sigma_true<-Lambda%*%t(Lambda)+0.1*diag(S) #We obtain Sigma. Here
      sigma_epsilon^2 is 0.1


  #We obtain the correlation matrix R
  R_true=cov2cor(Sigma_true)

  #We sample V from the model
  Y_cont<-L+rmvnorm(n = n, mean=rep(0,S), sigma=R_true)
  # We obtain our binary dataset. Secondo me vi conviene prima
     testare il modello con Y_cont e poi su Y
  Ybin<- ifelse(Y_cont>0,1,0)
  Ycont<-Y_cont
  give_back<-list(A_true = Lambda, B_true=B, Xdesign=X, mu_true=L,
     R_true=R_true, V=Ycont, Y=Ybin)
  return(give_back)
}
data<-simulation_fun()
```

Code 6: Function for simulating data

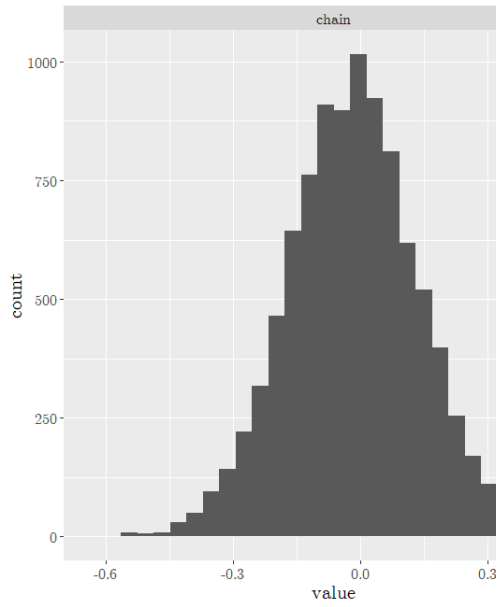
5.1 Convergence of the chain



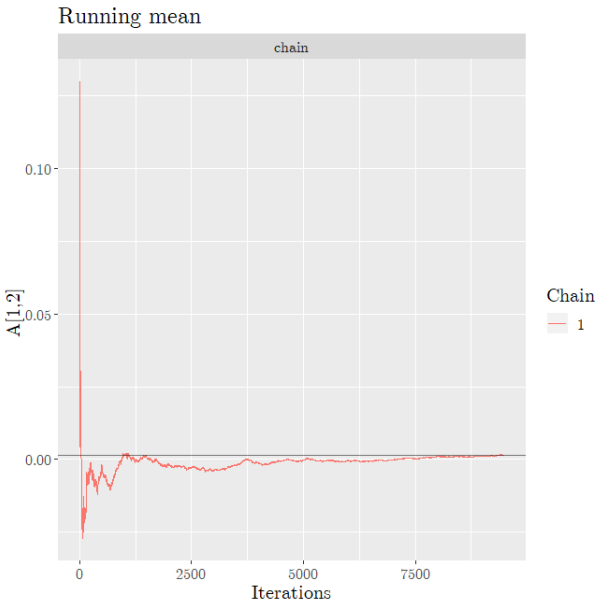
(a) Traceplot.



(b) Autocorrelation



(a) Values



(b) Running mean

Many elements of the chains (matrices in most cases) have been critically analyzed, using the usual techniques for studying its convergence. Here above, for instance, we show some of these plots for one element of matrix $\boldsymbol{A}$.

In order to estimate the correct behaviour of our model and code, it is widely more interesting computing credible regions for parameters. For this purpose we have built the function here below which creates a binary matrix, success and fail, whenever our posterior draws are or are not in the 0.95 credible region of true parameters.

```
data<-simulation_fun()
x<-data$Xdesign #x matrix of covariates
Y<-data$Y #Binary matrix Y

#Call of the function (R or Rcpp)
#result<-gjam_gibbs_sampler_R(alpha0,ndraws,burnin,N_stick,r,S, n_
   sites,x,Y)
result<-gjam_gibbs_sampler_Rcpp(alpha0,ndraws,burnin,N_stick,r,S, n
   _sites,x,Y)

mat_bin=matrix(0,nrow=S,ncol=r)
check_CR(result$A_inf, result$A_sup, data$A_true, mat_bin)

check_CR<-function(mat_inf, mat_sup, mat_true, mat_bin){
S<-dim(mat_bin)[1]
r<-dim(mat_bin)[2]

  for(i in 1:S){
    for(j in 1:r){
      if(mat_inf[i,j]<=mat_true[i,j] && mat_sup[i,j]>=mat_true[i,j]
          ){
        mat_bin[i,j]=1
      }
    }
  }
  mat_bin
}
```

Code 7: Definition and usage of credible regions check

Our simulations show that the Gibbs sampler works up to 80% in parameters estimation which is a really good result.

5.2 Computational time

First of all we have compared the time consuming of our code with respect to the one of `gjam` library. The table here below shows that we are able to speed up computations up to 2.3 times better:

S	n	r	$N_{stick} = \min(150, S)$	**time [s]**	
				Taylor&al	Rcpp
150	100	4	150	644.344	274.189
50	100	4	50	203.803	84.918
30	100	4	30	100.205	40.082
20	100	4	20	87.177	39.626

Moreover, we have analyzed computational time required by each singular step during the whole simulation of the sampler.

Here below the results choosing 3000 draws, 150 species, 150 truncation number of stick breaking process, 100 sites, 4 latent factors and mass parameter of Dirichlet Process equal to 1 (13.81 min).

Step 1	Step 2	Step 3	Step 4	Step 5	Step 6	Step 7	Step 8
2.247	34.219	167.664	5.506	1.346	0.703	569.358	47.714

6 Conclusion

During this project we had two main objectives since the very first beginning. The first was to work on the model part by choosing a not improper prior for the matrix of coefficients $\boldsymbol{B}$, by exploiting the conjugancy property we obtained a satisfying result. The second (and biggest) part was to obtain a faster Gibbs Sampler than the one given by Taylor-Rodríguez et al. (2017). Initially we where requested to implement it in `R`, but lot of computational errors concerning matrix inversions and few log probabilities came frequently out as the dimension of the problem increased. So the necessity of implementing the same Gibbs Sampler in Rcpp was getting more and more necessary.
At this point we focused on writing the Rcpp code relying on useful `C++` libraries such as Armadillo and Eigen. Comparing our results with the ones given by Gjam library we can finally say that we are satisfied of our work.

References

[BPD14] Anirban Bhattacharya, Debdeep Pati, and David Dunson. "Anisotropic function estimation using multi-bandwidth Gaussian processes". In: *Ann. Statist.* 42.1 (Feb. 2014), pp. 352–381. DOI: 10.1214/13-AOS1192. URL: https://doi.org/10.1214/13-AOS1192.

[CG98] S. Chib and E. Greenberg. "Analysis of Multivariate Probit Models". In: *Biometrika* 85 (June 1998), pp. 347–361. DOI: 10.1093/biomet/85.2.347.

[Cla+17] James S. Clark et al. "Generalized joint attribute modeling for biodiversity analysis: Median-zero, multivariate, multifarious data." In: *Ecological Monographs* (2017). URL: http://sites.nicholas.duke.edu/clarklab/code/.

[LKJ09] Daniel Lewandowski, Dorota Kurowicka, and Harry Joe. "Generating random correlation matrices based on vines and extended onion method". In: *Journal of Multivariate Analysis* 100 (Oct. 2009), pp. 1989–2001. DOI: 10.1016/j.jmva.2009.04.008.

[Ova+17] Otso Ovaskainen et al. "How are species interactions structured in species-rich communities? A new method for analysing time-series data". In: *Proceedings of the Royal Society B: Biological Sciences* 284 (May 2017), p. 20170768. DOI: 10.1098/rspb.2017.0768.

[Pol+14] Laura J. Pollock et al. "Understanding co-occurrence by modelling species simultaneously with a Joint Species Distribution Model (JSDM)". In: *Methods in Ecology and Evolution* 5.5 (2014), pp. 397–406. DOI: 10.1111/2041-210X.12180. eprint: https://besjournals.onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.12180. URL: https://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.12180.

[Tay+17] Daniel Taylor-Rodríguez et al. "Joint Species Distribution Modeling: Dimension Reduction Using Dirichlet Processes". In: *Bayesian Anal.* 12.4 (Dec. 2017), pp. 939–967. DOI: 10.1214/16-BA1031. URL: https://doi.org/10.1214/16-BA1031.

[Tik+17] Gleb Tikhonov, Nerea Abrego, et al. "Using joint species distribution models for evaluating how species-to-species associations depend on the environmental context". In: *Methods in Ecology and Evolution* 8 (Apr. 2017), pp. 443–452. DOI: 10.1111/2041-210X.12723.

[TO17] Gleb Tikhonov and Otso Ovaskainen. "Making more out of ecological community data: conceptual framework and its implementation as models and software". In: (Dec. 2017).