Bayesian Multi-Object Response Regression

Rene Gutierrez Marquez

2023-08-17

Contents

1 About

1.1 Package Description

This work documents the package `bmrr` introduced in “Multi-object Data Integration in the Study of Primary Progressive Aphasia”. It also presents an usage example in the context of the simulations on the paper for easy replication. BMRR stands for Bayesian Multi-Object Response Regression and deals with a scalar variable of interest as a dependent variable and two objects as responses, one a Network object and another an image object or more generally a Voxel object. Both objects are linked through regions of interest. The main purpose of the model is to determine which regions of interest are associated with the variable of interest. The model can incorporate other explanatory variables.

1.2 Package Requierments

The package requires the `mvtnorm` [Genz et al., 2021, Genz and Bretz, 2009] package.

1.3 Package Installation

To install the package enter:

```
library(devtools)
install_github("Rene-Gutierrez/bmrr")
```

1.4 Document Structure

In section 2 we introduce the model and variables

2 Model Framework

The main function of the package is `bmrr_sampler` which obtains samples according to the method described on the paper in section 3. For every observation $i \in \{1, \dots, n\}$:

- $y_i \in \mathbb{R}$: An scalar variable of interest.
- $\mathbf{A}_i \in \mathbb{R}^{P \times P}$ A network object, represented as a symmetric matrix of P by P, where we disregard the diagonal entries.
- $\mathbf{g}_{i,p} \in \mathbb{R}^{V_p}$ for $p \in 1, \dots, P$ voxel elements referenced by the same P regions as the Network object. Notice that we allow for the voxel elements to vary in size across the P regions.
- $\mathbf{x}_i \in \mathbb{R}^H$ A vector of other covariates that can be taken into consideration.

Then we consider the following linear models:

$$a_{i,(p,p)} = \sum_{h=1}^{H} \psi^a_{p,h} \psi^a_{p',h} x_{i,h} + \theta_{p,p'} y_i + e_i^{(p,p')} \quad p<p' \tag{1}$$

$$g_{i,(v,p)} = \sum_{h=1}^{H} \psi^g_{p,h} x_{i,h} + \beta_{v,p} y_i + w_i^{(p,p')} \quad p \in 1 < \dots, P, \quad v \in 1, \dots, V_p \tag{2}$$

, where

$$e_i^{(p,p')} \overset{iid}{\sim} N(0, \tau_\theta^2) \tag{3}$$

$$w_i^{(v,p)} \overset{iid}{\sim} N(0, \tau_\beta^2) \tag{4}$$

We can also represent this regression equations in stacked format as follows:

$$\mathbf{A}_i = \sum_{h=1}^{H} {}^{a}_{p,h} x_{i,h} + \ y_i + E_i \tag{5}$$

$$\mathbf{g}_{i,p} = \sum_{h=1}^{H} \mathbf{1}_{V_p} \psi^{g}_{h,p} x_{i,h} + \ {}_p y_i + \mathbf{w}_{i,p} \tag{6}$$

where we have stacked the elements as follows:

$$\begin{gathered}
\mathbf{A}_i \in \mathbb{R}^{P\times P}, \quad [\mathbf{A}_i]_{p,p'} = [\mathbf{A}_i]_{p',p} = a_{i,(p,p')} \quad \text{for} \quad p < p', \quad [\mathbf{A}_i]_{p,p} = 0 \quad \text{for} \quad p = p' \\
\in \mathbb{R}^{P\times P}, \quad [\ {}^{a}]_{p,p'} = [\ {}^{a}]_{p',p} = \psi^{a}_{p,h}\psi^{a}_{p',h}, \quad \text{for} \quad p < p', \quad [\ {}^{a}]_{p,p} = 0 \quad \text{for} \quad p = p' \\
\\
\in \mathbb{R}^{P\times P}, \quad [\ ]_{p,p'} = [\ ]_{p',p} = \theta_{p,p'} \quad \text{for} \quad p < p', \quad [\ ]_{p,p} = 0 \quad \text{for} \quad p = p' \\
\mathbf{E}_i \in \mathbb{R}^{P\times P}, \quad [\mathbf{E}_i]_{p,p'} = [\mathbf{E}_i]_{p',p} = a_{i,(p,p')} \quad \text{for} \quad p < p', \quad [\mathbf{E}_i]_{p,p} = 0 \quad \text{for} \quad p = p' \\
\mathbf{g}_{i,p} \in \mathbb{R}^{V_p}, \quad \mathbf{g}_{i,p} = (g_{i,1,p}, \dots, g_{i,V_p,p}) \quad \text{for} \quad p \in \{1, \dots, P\} \\
\mathbf{1}_{V_p} \in \mathbb{R}^{V_p}, \quad \text{A vector of ones.} \\
\mathbf{w}_{i,p} \in \mathbb{R}^{V_p}, \quad \mathbf{w}_{i,p} = (w_{i,1,p}, \dots, w_{i,V_p,p}) \quad \text{for} \quad p \in \{1, \dots, P\}
\end{gathered}$$

The main focus of analysis are the $\mathbf{A}_i$, $\mathbf{g}_{i,p}$ and y_i elements. Objects $\mathbf{A}_i$ and $\mathbf{g}_{i,p}$ are related through the regions p. This connection is developed further in the prior distributions.

The prior for the Network coefficients is set as follows:

$$\theta_{p,p'} | \lambda_{p,p'}, \tau_\theta^2, \sigma_\theta, \xi_p, \xi_{p'} \overset{ind.}{\sim} \xi_p \xi_{p'} N(0, \tau_\theta^2 \sigma_\theta^2 \lambda_{p,p'}^2) + (1 - \xi_p \xi_{p'}) \delta_0 \quad p < p') \tag{7}$$

while the prior for the Voxel coefficients is:

$${}_{v,p} | \phi_{v,p}^2, \eta_p^2, \tau_\beta^2, \xi_p \overset{ind.}{\sim} \xi_p N(0, \tau_\beta^2 \eta_p^2 \phi_{v,p}^2)\ {}_p + (1 - \xi_p) \delta_0 \quad v \in \{1, \dots, V_p\} \quad p \in \{1, \dots, P\} \tag{8}$$

That is, a region specific Spike and Slab prior on the Network and Voxel coefficients is applied, that simultaneously determines which regions are relevant through the spike indicators ξ_p [George and McCulloch, 1997]. The spike component of the the coefficients is given a Horseshoe structure [Carvalho et al., 2010]. Then the Horseshoe structure for the Network coefficients is set as:

$$\begin{gathered}
\sigma_\theta \sim C^+(0, 1) \\
\lambda_{p,p'} \overset{iid}{\sim} C^+(0, 1) \quad p \in \{1, \dots, P\}
\end{gathered}$$

The Horseshoe structure for the Voxel Coefficients is given by:

$$\eta_p^2 \overset{iid}{\sim} C^+(0, 1) \quad p \in \{1, \dots, P\}$$

$$\phi_{v,p} \overset{iid}{\sim} C^{+}(0,1) \quad v \in \{1,\dots,V_p\} \quad p \in \{1,\dots,P\}$$

For the spike indicators, the prior is set as:

$$\xi_p \overset{iid}{\sim} Ber(\nu)$$

$$\nu \sim Beta(a_\nu, b_\nu)$$

where a Beta prior is assigned to the Bernoulli prior probability to account for multiplicity correction.

Inverse-Gamma priors are assigned to the error variances:

$$\tau_\theta^2 \sim IG(a_{\tau_\theta}, b_{\tau_\theta})$$

$$\tau_\beta^2 \sim IG(a_{\tau_\beta}, b_{\tau_\beta})$$

Finally, for the other covariate coefficients:

$$\psi^a_{p,h}, \psi^g_{p,h} \propto 1 \quad p \in \{1,\dots,P\}, \quad h \in \{1,\dots,H\}$$

3 Posterior Computation

Bayesian estimation of the model is performed through Gibbs sampling which cycles through the following steps:

1. For each $p \in \{1,...,P\}$ sample from $\psi^a_{p,.}$, from the full conditional $p(\psi^a_{p,.}|\psi^a_{-p,.}, \Theta, \tau^2_\theta, A)$.
2. For each $p \in \{1,...,P\}$ sample from $\psi^g_{p,.}$, from the full conditional $p(\psi^g_{p,.}|\beta_p, \tau^2_\beta, g_p)$.
3. For each $p \in \{1,...,P\}$ sample jointly ξ_p, $\Theta_{-p,p}$ and β_p from:

$$\begin{aligned} p(\xi_p, \Theta_{-p,p}, \beta_p | \nu, \tau^2_\theta, \tau^2_\beta, \sigma^2_\theta, \eta^2_p, \Lambda_{-p,p}, \phi^2_p, A, G) \\ = p(\Theta_{-p,p}, \beta_p | \xi, \tau^2_\theta, \tau^2_\beta, \sigma^2_\theta, \eta^2_p, \Lambda_{-p,p}, \phi^2_p, A, G) \\ \times\, p(\xi_p | \nu, \xi_{-p}, \tau^2_\theta, \tau^2_\beta, \sigma^2_\theta, \eta^2_p, \Lambda_{-p,p}, \phi^2_p, A, G) \end{aligned}$$

4. Sample τ^2_θ from the full conditional $p(\tau^2_\theta|\xi, \Theta, \sigma^2_\theta, \Lambda, A)$
5. Sample τ^2_β from the full conditional: $p(\tau^2_\beta|\xi, \beta_1, ..., \beta_P, \eta^2_1, ..., \eta^2_P, \phi^2_1, ..., \phi^2_P, G)$
6. Sample ν from the full conditional $p(\nu|\xi)$.
7. Sample the horseshoe parameters using the latent variable approach as in [Makalic and Schmidt, 2016].

3.1 Full Conditional for $\psi^a_{p,.}$

$$\psi^a_{p,.}|\psi^a_{-p,.}, \Theta, \sigma^2, A \sim N\left(\hat{\psi}^a_{p,.}, \tau^2_\theta (W'W)^{-1}\right)$$

where

$$\begin{aligned} \hat{\psi}^a_{p,.} &= (W'W)^{-1}W'R \\ W &= [W'_1, ..., W'_{p-1}, W'_{p+1}, \dots, W'_P]' \\ W_q &= (\psi^a_{q,.} \otimes 1_n) \cdot X \quad q \neq p \\ R &= \text{vec}(A_{.,-p,p}) - \Theta_{-p,p} \otimes y \end{aligned}$$

3.2 Full conditional for $\psi^g_{p,.}$

$$\psi^g_{p,.}|\beta_p, \sigma^2, g_p \sim N\left(\hat{\psi}^g_{p,.}, \tau^2_\beta (W'W)^{-1}\right)$$

where

$$\begin{aligned} \hat{\psi}^g_{p,.} &= (W'W)^{-1}W'R \\ W &= 1_{V_p} \otimes x \\ R &= (R'_{.,v,p}, ..., R'_{.,V_p,p})' \\ R_{.,v,p} &= g_{.,v,p} - \beta_{v,p} y \quad \forall v \in \{1, ..., V_p\} \end{aligned}$$

3.3 Full conditional for ξ_p, $\Theta_{-p,p}$ and β_p

To sample $\Theta_{-p,p}, \beta_p$ set them to 0 if $\xi_p = 0$, if $\xi = 1$ then there are 2 cases.

If $\xi_{p'} = 0 \quad \forall p'$ then set $\Theta_{-p,p} = 0$ and sample β_p from:

$$\beta_p|\tau^2_\beta, \eta^2_p, \phi^2_p, G \sim N(\hat{b}, \tau^2_\beta \text{diag}(1/(S^2_{yy} + 1/L)))$$

where

$$\begin{aligned} \hat{b} &= S_{xy}/(S^2_{yy} + 1/L) \\ S_{xy} &= S_{gy,p} \\ S^2_{yy} &= y'y \\ L &= \eta^2_p(\phi^2_p) \\ (S_{gy,p})_v &= \sum_{i=1}^{n} R^g_{i,v,p} y_i \\ R^g_{i,v,p} &= g_{i,(v,p)} - \sum_{h=1}^{H} \psi^g_{p,h} x_{i,h} \end{aligned}$$

And sample ξ_p from

$$\xi_p|\nu, \tau^2_\beta, \eta^2_p, \phi^2_p, G \sim Bernoulli(\hat{v}_p)$$

where

$$\begin{aligned} \hat{v}_p &= o_p/(1 + o_p) \\ o_p &= \exp(\sum_{i=1} c_i) \frac{\nu}{1 - \nu} \\ c_i &= -\frac{1}{2}\left(\log(L_i) + \log\left(S^2_{yy} + \frac{1}{L_i}\right)\right) + \hat{b}^2_i \frac{S^2_{yy} + \frac{1}{L_i}}{2\tau^2_\beta} \end{aligned}$$

where L_i and $\hat{b}$ are as in the sampling of β_p.

If $\exists p' \ni \xi_p \neq 0$, then sample $\Theta_{-p,p}$ and β_p from:

$$\Theta_{-p,p}, \beta_p | \xi_{-p}, \tau_\beta^2, \tau_\theta^2 \sigma_\theta^2, \eta_p^2, \Lambda_{-p,p}, \phi_p^2, A, G \sim N(\hat{b}, \text{diag}(\tau/(S_{yy}^2 + 1/L)))$$

where

$$\begin{aligned}
\hat{b} &= S_{xy}/(S_{yy}^2 + 1/L) \\
S_{xy} &= [(S_{ay})_{p,-p}[\xi_{-p} = 1]', S'_{gy,p}]' \\
S_{yy}^2 &= y'y \\
\tau &= (\tau_\theta^2 1_P[\xi_{-p} = 1]', \tau_\beta^2 1'_{V_p})' \\
L &= [\sigma_\theta^2 \Lambda_{p,-p}[\xi_{-p} = 1]', \eta_p^2(\phi_p^2)']' \\
(S_{ay})_{p,p'} &= \sum_{i=1}^{n} (R_i^a)_{p,p'} y_i \\
(S_{gy,p})_v &= \sum_{i=1}^{n} R_{i,v,p}^g y_i \\
(R_i^a)_{p,p'} &= (A_i)_{p,p'} - \sum_{h=1}^{H} (\psi^a)_{p,h} (\psi^a)_{p',h} x_{i,h} \\
R_{i,v,p}^g &= g_{i,(v,p)} - \sum_{h=1}^{H} \psi_{p,h}^g x_{i,h}
\end{aligned}$$

We sample ξ_p from

$$\xi_p | \nu, \xi_{-p}, \tau_\theta^2, \tau_\beta^2, \sigma_\theta^2, \eta_p^2, \Lambda_{-p,p}, \phi_p^2, A, G \sim Bernoulli(\hat{v}_p)$$

where

$$\begin{aligned}
\hat{v}_p &= o_p/(1 + o_p) \\
o_p &= \exp(\sum_{i=1} c_i) \frac{\nu}{1 - \nu} \\
c_i &= -\frac{1}{2} \left(\log(L_i) + \log \left(S_{yy}^2 + \frac{1}{L_i} \right) \right) + \hat{b}_i \frac{S_{yy}^2 + \frac{1}{L_i}}{2\tau^2}
\end{aligned}$$

where L_i and $\hat{b}$ and τ are as in the sampling of $\Theta_{-p,p}, \beta_p$.

3.4 Full Conditional for τ_θ^2

$$\tau_\theta^2 | \xi, \Theta, \sigma_\theta^2, \Lambda, A \sim IG(\hat{a}_{\tau_\theta}, \hat{b}_{\tau\theta})$$

where

$$
\begin{aligned}
\hat{b}_{\tau_\theta} &= b_{\tau_\theta} + \frac{\sum_{i=1}^{n}\sum_{p<p'}(R_i^a)_{p,p'}}{2} \\
&+ \frac{\sum_{p<p'}\Theta^2_{\xi=1,\xi=1}/\Lambda_{\xi=1,\xi=1}/\sigma^2_\theta}{2} \\
\hat{a}_{\tau_\theta} &= a_{\tau_\theta} + \frac{\frac{np(p-1)}{2} + \frac{nq(q-1)}{2}}{2} \\
R_i^a &= \left(A_i - \sum_{h=1}^{H}\psi_h^a(\psi_h^a)' x_{i,h} - \Theta\, y_i\right)^2
\end{aligned}
$$

3.5 Full Conditional for τ^2_β

$$
\tau^2_\beta|\xi, \beta_1, ..., \beta_P, \eta^2_1, ..., \eta^2_P, \phi^2_1, ..., \phi^2_P, G \sim IG(\hat{a}_{\tau_\beta}, \hat{b}_{\tau_\beta})
$$

where

$$
\begin{aligned}
\hat{b}_{\tau_\beta} &= b_{\tau_\beta} + \frac{\sum_{i=1}^{n}\sum_{p=1}^{P} R^g_{i,p} + \sum_{p=1}^{P}\xi_p\beta^2_p/(\eta_p\phi^2_p)}{2} \\
\hat{a}_{\tau_\beta} &= a_{\tau_\beta} + \frac{n\sum_{p=1}^{P} V_p + n\sum_{p=1}^{P} V_p\xi_p}{2} \\
R^g_{i,p} &= \left(g_{i,p} - \sum_{h=1}^{H} 1_{V_p}\psi^g_{p,h}x_{i,h} - \beta_p\, y_i\right)^2
\end{aligned}
$$

3.6 Full conditional ν

$$
\nu|\xi_1, \dots, \xi_p \sim Beta\left(a_\nu + \sum_{p=1}^{P}\xi_p, b_\nu + P - \sum_{p=1}^{P}\xi_p\right)
$$

3.7 Full Conditional for the Horseshoe Structure

Following [Makalic and Schmidt, 2016] auxiliary variables are used for the priors of the $\lambda_{p,p'}$'s, σ^2_θ, $\phi_{v.p}$'2 and η_p's as follows:

$$
\begin{aligned}
\lambda^2_{p,p'} &\sim IG(1/2, 1/\kappa_{\lambda_{p,p'}}) \\
\sigma^2_\theta &\sim IG(1/2, 1/\kappa_{\sigma^2_\theta}) \\
\phi^2_{v,p} &\sim IG(1/2, 1/\kappa_{\phi_{v,p}}) \\
\eta^2_p &\sim IG(1/2, 1/\kappa_{\eta_p}) \\
\kappa_{\lambda_{p,p}}, \kappa_{\sigma_\theta}, \kappa_{\phi_{v,p}}, \kappa_{\eta_p} &\overset{iid}{\sim} IG(1/2, 1) \quad v \in \{1, \dots, V_p\} \quad p \in \{1, \dots, P\}
\end{aligned}
$$

Then the full conditionals for the Network Horseshoe parameters (including auxiliary variables) are given by

$$
\lambda^2_{p,p'}|\sigma^2_\theta, \tau^2_\theta, \theta_{p,p'}, \xi_p, \xi_{p'} \overset{ind.}{\sim} IG\left(\frac{1}{2} + \frac{\xi_p\xi_{p'}}{2}, \frac{1}{\kappa_{\lambda_{p,p'}}} + \frac{\xi_p\xi_{p'}\theta^2_{p,p'}}{2\sigma^2_\theta\tau^2_\theta}\right) \quad p < p' \tag{9}
$$

$$\sigma_\theta^2|\Lambda,\tau_\theta^2,\Theta,\xi_p \overset{ind.}{\sim} IG\left(\frac{1}{2}+\frac{\sum_{p<p'}\xi_p\xi_{p'}}{2}, \frac{1}{\kappa_{\sigma_\theta^2}}+\frac{1}{2\tau_\theta^2}\sum_{p<p'}\frac{\xi_p\xi_{p'}\theta_{p,p'}^2}{\lambda_{p,p'}^2}\right) \tag{10}$$

$$\kappa_{\lambda_{p,p'}}|\lambda_{p,p'}^2 \sim IG\left(1, 1+\frac{1}{\lambda_{p,p'}^2}\right) \tag{11}$$

$$\kappa_{\sigma_\theta}|\sigma_\theta^2 \sim IG\left(1, 1+\frac{1}{\sigma_\theta^2}\right) \tag{12}$$

And the full conditionals for the voxel Horseshoe parameters (including auxiliary variables) are given by

$$\tag{13}$$

$$\phi_{v,p}^2|\eta_p^2,\tau_\beta^2,\phi_{v,p},\xi_p \overset{ind.}{\sim} IG\left(\frac{1}{2}+\frac{\xi_p}{2}, \frac{1}{\kappa_{\phi_{v,p}}}+\frac{\xi_p\beta_{v,p}^2}{2\eta_p^2\tau_\beta^2}\right) \quad v\in\{1,\dots,V_p\} \quad p\in\{1,\dots,P\} \tag{14}$$

$$\eta_p^2|\Phi,\tau_\beta^2,B,\xi_p \overset{ind.}{\sim} IG\left(\frac{1}{2}+\frac{V_p\xi_p}{2}, \frac{1}{\kappa_{\eta_P}}+\frac{\xi_p}{2\tau_\beta^2}\sum_{v=1}^{V_p}\frac{\beta_{v,p}^2}{\phi_{v,p}^2}\right) \quad p\in\{1,\dots,P\} \tag{15}$$

$$\kappa_{\phi_{v,p}}|\phi_{v,p}^2 \sim IG\left(1, 1+\frac{1}{\phi_{v,p}^2}\right) \tag{16}$$

$$\kappa_{\eta_p}|\eta_p^2 \sim IG\left(1, 1+\frac{1}{\eta_p^2}\right) \quad p\in\{1,\dots,P\} \tag{17}$$

4 Functions

In this section the package functions are explained (including those that are not available to use to the user).

In this section variable names in the code are written in code mode while the variables described in Model Framework are written in math mode, for example in the code the main variable is `y` while in model the main variable is written y. Unfortunately, the name of the variables in the code and the model framework are not always a close match as in the case of `y` and y. The corresponding relations will be indicated when each variable is introduced in the code.

4.1 `bmrr_sampler`

4.1.1 Desription

The main function of the package is `bmrr_sampler`. It is in charge of receiving the data and model set-up and returns samples for all the model parameters, as well as the last state of the variables. The first part of the function deals with the set-up, while the second part loops through the function `bmrr_iterator` to obtain and save the samples. The actual computation of the full conditionals is done by `bmrr_iterator`.

4.1.2 Inputs

`bmrr_sampler` receives three types of inputs, data, hyper-parameters and set-up inputs.

The data inputs required by the function. They are as follows:

Code	Model	Description
`y`	y	This is the variable of interest. It should be inputted as a vector of size n, the number of observations.
`A`	A	The Network object. It should be inputted as an array of size $(n \times P \times P)$. For each observation i, `A[i,,]` must be symmetric with `NA` as diagonal elements.
`G`	$\{g_p\}_{p=1}^P$	The voxel object. It should be inputted as an array of size $(n \times V \times P)$, where $V = \max(\{V_p\}_{p=1}^P)$. If the regions are not the same size (i.e. there is a p for which $V_p < V$), then pad the entry `G[i,p,]` with `NA`'s, do not use `0` or any other numeric value.
`X`	X	Other covariates to take in consideration. It should be inputted as a matrix of size $(n \times H)$. Each column `X[,h]` represents one covariate.

The hyper-parameter inputs are not required, if they are not specified the default values are used. They are as follows:

Code	Model	Description
`a_sT`	a_{τ_θ}	Shape hyper-parameter of the Inverse-gamma prior for τ_θ^2. An scalar. By default, `a_sT = 1`.
`b_sT`	b_{τ_θ}	Scale hyper-parameter of the Inverse-gamma prior for τ_θ^2. An scalar. By default, `b_sT = 1`.
`a_sB`	a_{τ_β}	Shape hyper-parameter of the Inverse-gamma prior for τ_β^2. An scalar. By default, `a_sB = 1`.
`b_sB`	b_{τ_β}	Scale hyper-parameter of the Inverse-gamma prior for τ_β^2. An scalar. By default, `b_sB = 1`.
`a_nu`	a_ν	Shape 1 hyper-parameter of the Beta prior for ν. An scalar. By default, `a_sB = 1`.
`b_nu`	b_ν	Shape 2 hyper-parameter of the Beta prior for ν. An scalar. By default, `b_sB = 1`.

The set-up inputs also have set defaults, however, unlike the hyper-parameter inputs, the set-up inputs are might require more attention from the user as they deal with the characteristics of the MCMC samples. As set-up inputs, they do not have a model equivalent.

They are as follows:

Code	Description
`nmcmc`	Number of MCMC samples of the output. Default is 1000. Notice this number, does not include the samples that are lost due to burn-in and thinning.
`burnin`	Number of initial iterations of the MCMC to discard. Default is 0, that is there is no burn-in.
`thinning`	Number of iterations of the MCMC chain to obtain one output sample.
`state`	A list of containing all the model parameters. It can be used in two ways. The first one is to initialize the MCMC at a selected starting point. The second one is to continue a MCMC sample chain. It is also an output of the function, so it is readily available in case the function is used sequentially.

`small_out`	A Boolean that indicates if a small set of the parameters of the model is required for memory purposes. By default `small_out = FALSE`, which returns all the model parameters. If `small_out = TRUE`, then only the first order parameters in the hierarchy and the indicators are returned, that is the parameters that appear on equations (1), (2), (3), (4) and ξ. Further more the network and voxel coeeficients are returned in a single object.

4.1.3 Output

The function returns a list with two elements. The first one, and most important is a list called `sam` that contains `nmcmc` samples of the model parameters after discarding burn-in iterations according to `burnin` and skipping iterations according to `thinning`.

For example, if:

- `nmcmc = 100`
- `burnin = 10`
- `thinning = 3`

the function will perform `10 + 100 * 3 = 310` MCMC iterations to return `100` samples for each parameter, as specified by `small_out`.

The second output of the function is `state` a list all the variables on the last iteration of the MCMC.

If `small_out = FALSE` (by default) samples for every model parameter and auxiliary variables are returned.

Code	Model	Description
`g`	ξ	Samples for the region indicator variables. It is returned as a binary matrix of size `nmcmc` rows by P. For sample s, `g[s,p]` $= {\xi_p^s}'$.
`nu`	ν	Samples of the probability parameter of the Bernoulli prior of the indicators ξ_p. It is as vector of size `nmcmc`. For sample s, `nu[s]`$= \nu^s$.
`Theta`	Θ	Samples for the Network coefficients. It is an array of size (`nmcmc`$\times P \times P$). For sample s, `Theta[s,p,q]`$= (\Theta^s)_{p,q} = \theta^s_{p,q}$ for $p \neq q$ and `Theta[s,p,p]=NA`.
`B`	$\{g_p\}_{p=1}^P$	Samples for the Voxel coefficients. It is an array of size (`nmcmc`$\times V \times P$), where $V = \max\{V_p\}_{p=1}^P$. For sample s, `B[s,v,p]`$= \beta_{v,p}$ for $v \leq V_p$, and `B[s,v,p]=NA` for $v > V_p$.
`DA`	$\{\psi^a_{p,h}\}_{p=1,h=1}^{P,H}$	Samples of the bilinear structure of the Network equation coefficients of the covariates X. It is an array of size (`nmcmc` $\times P \times H$). For sample s, `DA[s,p,h]`$= {\psi^{a,s}_{p,h}}'$.
`DG`	$\{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$	Samples of the structure of the Voxel equation coefficients of the covariates X. It is an array of size (`nmcmc` $\times P \times H$). For sample s, `DA[s,p,h]`$= {\psi^{g,s}_{p,h}}'$.
`sT2`	τ^2_θ	Samples of the variance of the error of the Network equation. It is a vector of size `nmcmc`. For sample s, `sT2[s]` $= {\tau^2_\theta}^s$.
`sB2`	τ^2_β	Samples of the variance of the error of the Voxel equation. It is a vector of size `nmcmc`. For sample s, `sB2[s]` $= {\tau^2_\beta}^s$.
`t2T`	σ^2_θ	Samples of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a vector of size `nmcmc`. For sample s, `t2T[s]`$= {\sigma^2_\theta}^s$.
`l2T`	Λ	Samples of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is an array of size `nmcmc`$\times P \times P$. For sample s, `l2T[s,p,q]`$= (\Lambda)^s_{p,q} = \lambda^s_{p,q}$ for $p \neq q$ and `l2T[s,p,p]=NA`.

`xiT`	$\kappa_{\sigma^2_\theta}$	Samples of the auxiliary variable of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a vector of size `nmcmc`. For sample s, `xiT[s]`$= \kappa^s_{\sigma^2_\theta}$.
`vT`	$\kappa_{\lambda_{p,p'}}$	Samples of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is an array of size `nmcmc`$\times P \times P$. For sample s, `vT[s,p,q]`$= (\Lambda)^s_{p,q} = \kappa^s_{\lambda_{p,q}}$ for $p \neq q$ and `l2T[s,p,p] = NA`.
`t2B`	$\{\eta_p\}_{p=1}^P$	Samples of the global shrinking parameters for the Horseshoe structure of the Voxel coefficients. It is a matrix of size `nmcmc`$\times P$. For sample s, `t2B[s,p]`$= {\eta_p}^s$.
`l2B`	$\{\phi_{v,p}\}_{v=1,p=1}^{V_p,P}$	Samples of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is an array of size `nmcmc`$\times V \times P$. For sample s, `l2B[s,v,p]`$= \phi^s_{v,p} = \phi^s_{v,p}$ for $v \leq V_p$ and `l2B[s,v,p]=NA` for $v > V_p$.
`xiB`	κ_{η_p}	Samples of the auxiliary variables of the global shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a matrix of size `nmcmc`$\times P$. For sample s, `xiT[s,p]`$= \kappa^s_{\eta_p}$.
`vB`	$\kappa_{\phi_{v,p}}$	Samples of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is an array of size `nmcmc`$\times V \times P$. For sample s, `vB[s,v,p]`$= \kappa^s_{\phi_{v,p}}$ for $v \leq V_p$ and `vB[s,v,p] = NA` for $v > V_p$.

If `small_out = TRUE` only the samples for the following parameters are returned:

Code	Model	Description
`g`	ξ	Samples for the region indicator variables. It is returned as a binary matrix of size `nmcmc` rows by P. For sample s, `g[s,p]` $= \xi_p^{s'}$.
`B`	Θ and $\{g_p\}_{p=1}^P$	Samples for the network and voxel coefficients. A matrix of size (`nmcmc` $\times P(P-1)/P + V$). The order of the coefficients is obtained by stacking the coefficients of the network object and then the coefficients for the voxel object. The coefficients of the network object are ordered by `upper.tri` applied to Θ and the coefficients of the voxel object are ordered by stacking the $\{\beta_p\}_{p=1}^P$ objects. That is each row of `B` is a sample s, such that `B[s,] =` $(\theta^s_{1,2}, \theta^s_{1,3}, \theta^s_{2,3}, \dots, \theta^s_{1,P}, \dots, \theta^s_{P-1,P}, {\beta_1^s}', \dots, {\beta_P^s}')$.
`DA`	$\{\psi^a_{p,h}\}_{p=1,h=1}^{P,H}$	Samples of the bilinear structure of the Network equation coefficients of the covariates X. It is an array of size (`nmcmc` $\times P \times H$). For sample s, `DA[s,p,h]`$= \psi^{a,s}_{p,h}$.
`DG`	$\{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$	Samples of the structure of the Voxel equation coefficients of the covariates X. It is an array of size (`nmcmc` $\times P \times H$). For sample s, `DA[s,p,h]`$= \psi^{g,s}_{p,h}$.
`sT2`	τ^2_θ	Samples of the variance of the error of the Network equation. It is a vector of size `nmcmc`. For sample s, `sT2[s]` $= {\tau^2_\theta}^s$.
`sB2`	τ^2_β	Samples of the variance of the error of the Voxel equation. It is a vector of size `nmcmc`. For sample s, `sB2[s]` $= {\tau^2_\beta}^s$.

The `state` output is a list containing all the model variables (including auxiliaries). Here are the elements of the list:

Code	Model	Description
`g`	ξ	Last MCMC iteration of the region indicator variables. It is returned as a binary vector of size P. `g[p]` $= \xi_p$.

`nu`	ν	Last MCMC iteration of the probability parameter of the Bernoulli prior of the indicators ξ_p. It is a scalar. `nu`$= \nu$.
`Theta`	Θ	Last MCMC iteration of the Network coefficients. It is a matrix of size $(\times P \times P)$. `Theta[p,q]`$= (\Theta)_{p,q} = \theta_{p,q}$ for $p \neq q$ and `Theta[p,p]=NA`.
`B`	$\{g_p\}_{p=1}^P$	Last MCMC iteration of the Voxel coefficients. It is a matrix of size $(\times V \times P)$, where $V = \max\{V_p\}_{p=1}^P$. `B[v,p]`$= \beta_{v,p}$ for $v \leq V_p$, and `B[v,p]=NA` for $v > V_p$.
`DA`	$\{\psi^a_{p,h}\}_{p=1,h=1}^{P,H}$	Last MCMC iteration of the bilinear structure of the Network equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^a_{p,h}$.
`DG`	$\{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$	Last MCMC iteration of the structure of the Voxel equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^g_{p,h}$.
`sT2`	τ^2_θ	Last MCMC iteration of the variance of the error of the Network equation. It is a scalar. `sT2[s]`$= \tau^2_\theta$.
`sB2`	τ^2_β	Last MCMC iteration of the variance of the error of the Voxel equation. It is a scalar. `sB2[s]`$= \tau^2_\beta$.
`t2T`	σ^2_θ	Last MCMC iteration of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `t2T`$= \sigma^2_\theta$.
`l2T`	Λ	Last MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is a matrix of size $\times P \times P$. `l2T[p,q]`$= (\Lambda)_{p,q} = \lambda_{p,q}$ for $p \neq q$ and `l2T[s,p,p]=NA`.
`xiT`	$\kappa_{\sigma^2_\theta}$	Last MCMC iteration of the auxiliary variable of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `xiT`$= \kappa_{\sigma^2_\theta}$.
`vT`	$\kappa_{\lambda_{p,p'}}$	Last MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is a matrix of size $\times P \times P$. `vT[p,q]`$= (\Lambda)_{p,q} = \kappa_{\lambda_{p,q}}$ for $p \neq q$ and `l2T[p,p] = NA`.
`t2B`	$\{\eta_p\}_{p=1}^P$	Last MCMC iteration of the global shrinking parameters for the Horseshoe structure of the Voxel coefficients. It is a vector of size P. `t2B[p]`$= \eta_p$.
`l2B`	$\{\phi_{v,p}\}_{v=1,p=1}^{V_p,P}$	Last MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a matrix of size $\times V \times P$. `l2B[v,p]`$= (\Phi)_{v,p} = \phi_{v,p}$ for $v \leq V_p$ and `l2B[v,p]=NA` for $v > V_p$.
`xiB`	κ_{η_p}	Last MCMC iteration of the auxiliary variables of the global shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a vector of size P. `xiT[p]`$= \kappa_{\eta_p}$.
`vB`	$\kappa_{\phi_{v,p}}$	Last MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients .It is a matrix of size $\times V \times P$. `vB[v,p]`$= \kappa_{\phi_{v,p}}$ for $v \leq V_p$ and `vB[v,p] = NA` for $v > V_p$.

4.1.4 Walkthrough

4.1.4.1 Problem Dimensions Obtains the sample size and dimensions of the objects.

- `N`$= n$. The number of observations.
- `mV`$V = \max\{V_p\}_{p=1}^P$. The maximum number of voxels per region.
- `P`$= P$. The number of regions.
- `H`$= H$. Number of other covariates (other than the main variable).

4.1.4.2 Creates Auxiliary Variables This variables are used on every iteration of the MCMC `bmrr_iterator`.

- `C`. Is a Boolean matrix of size $V \times P$ that indicates which entries of the voxel data are valid entries. For every observation `i`, `G[i,,][C]` $= g'_{i,1}, \cdots, g'_{i,P}$.
- `Zv` It is a transformation of the Network and Voxel data objects into one matrix. Each row is an observation of the Network object followed by the voxel object. Sometimes is convenient to work with the original data objects and sometimes is more convenient to work with the stacked object. `Zv[i,]`$= (a_{i,(1,2)}, a_{i,(1,3)}, a_{i,(2,3)}, \cdots, a_{i,(1,P)}, \cdots, a_{i,(P-1,P)}, g'_{i,1}, \cdots, g'_{i,P})$.
- `XG`. A replication of the covariates V times, stacked one over the other it is used for the computation of $\{\psi^g_{v,p}\}_{v=1,p=1}^{V_p,P}$.

4.1.4.3 Sample Holders Creates the holders of the samples according to `small_out`.

4.1.4.4 Initialization Initializes the MCMC, if `state` is provided, then it is used as the initial value otherwise default values are used.

4.1.4.5 Progress Bar Creates the progress bar to show the progress of the MCMC.

4.1.4.6 First Run Computes the first sample of the MCMC (it is discarded in every case).

4.1.4.7 Sampling Goes through the MCMC by calling `bmrr_iterator`, saves the samples after considering `thinning` and `burnin` as well as `small_out` and updates the progress bar.

4.1.4.8 Saves the Samples Saves the samples on a list according to `small_out`.

4.1.4.9 Saves the State Saves the last iteration of the MCMC in a list.

4.1.4.10 Returns Values Returns the two outputs, the MCMC samples (`sam`) and the last iteration of the MCMC (`state`).

4.2 `bmrr_iterator`

4.2.1 Description

This is the function that actually samples at each of the iterations of the MCMC. It receives as inputs the data (with the auxiliary data transformations of it), the hyper-parameters of the hyper-priors and the current state of the MCMC. It performs the sampling of all the model parameters except for the Network and Voxel coefficients, and region indicators. For theses parameters it relies on the `group_iterator` function.

4.2.2 Inputs

The data inputs are directly pass through the `bmrr_sampler` function with the addition of two useful transformations. The data inputs are as follows:

Code	Model	Description

`y`	y	This is the variable of interest. It should be inputted as a vector of size n, the number of observations.
`A`	A	The Network object. It should be inputted as an array of size $(n \times P \times P)$. For each observation i, `A[i,,]` must be symmetric with `NA` as diagonal elements.
`G`	$\{g_p\}_{p=1}^{P}$	The voxel object. It should be inputted as an array of size $(n \times V \times P)$, where $V = \max(\{V_p\}_{p=1}^{P})$. If the regions are not the same size (i.e. there is a p for which $V_p < V$), then pad the entry `G[i,p,]` with `NA`'s, do not use `0` or any other numeric value.
`X`	X	Other covariates to take in consideration. It should be inputted as a matrix of size $(n \times H)$. Each column `X[,h]` represents one covariate.
`Zv`		It is a transformation of the Network and Voxel data objects into one matrix. Each row is an observation of the Network object followed by the voxel object. `Zv[i,]`$= (a_{i,(1,2)}, a_{i,(1,3)}, a_{i,(2,3)}, \cdots, a_{i,(1,P)}, \cdots, a_{i,(P-1,P)}, g'_{i,1}, \cdots, g'_{i,P})$.
`XG`		A replication of the covariates V times, stacked one over the other it is used for the computation of $\{\psi^g_{v,p}\}_{v=1,p=1}^{V_p,P}$.
`C`		Is a Boolean matrix of size $V \times P$ that indicates which entries of the voxel data are valid entries. For every observation `i`, `G[i,,][C]`$= g'_{i,1}, \ldots, g'_{i,P}$.

The MCMC current state variables are:

Code	Model	Description
`g`	ξ	Current MCMC iteration of the region indicator variables. It is a binary vector of size P. `g[p]` $= \xi_p$.
`nu`	ν	Current MCMC iteration of the probability parameter of the Bernoulli prior of the indicators ξ_p. It is a scalar. `nu`$= \nu$.
`Theta`	Θ	Current MCMC iteration of the Network coefficients. It is a matrix of size $(\times P \times P)$. `Theta[p,q]`$= (\Theta)_{p,q} = \theta_{p,q}$ for $p \neq q$ and `Theta[p,p]=NA`.
`B`	$\{g_p\}_{p=1}^{P}$	Current MCMC iteration of the Voxel coefficients. It is a matrix of size $(\times V \times P)$, where $V = \max\{V_p\}_{p=1}^{P}$. `B[v,p]`$= \beta_{v,p}$ for $v \leq V_p$, and `B[v,p]=NA` for $v > V_p$.
`DA`	$\{\psi^a_{p,h}\}_{p=1,h=1}^{P,H}$	Current MCMC iteration of the bilinear structure of the Network equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^a_{p,h}$.
`DG`	$\{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$	Current MCMC iteration of the structure of the Voxel equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^g_{p,h}$.
`sT2`	τ^2_θ	Current MCMC iteration of the variance of the error of the Network equation. It is a scalar. `sT2[s]`$= \tau^2_\theta$.
`sB2`	τ^2_β	Current MCMC iteration of the variance of the error of the Voxel equation. It is a scalar. `sB2[s]`$= \tau^2_\beta$.
`t2T`	σ^2_θ	Current MCMC iteration of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `t2T`$= \sigma^2_\theta$.
`l2T`	Λ	Current MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is a matrix of size $\times P \times P$. `l2T[p,q]`$= (\Lambda)_{p,q} = \lambda_{p,q}$ for $p \neq q$ and `l2T[s,p,p]=NA`.
`xiT`	$\kappa_{\sigma^2_\theta}$	Current MCMC iteration of the auxiliary variable of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `xiT`$= \kappa_{\sigma^2_\theta}$.

`vT`	$\kappa_{\lambda_{p,p'}}$	Current MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is a matrix of size $\times P \times P$. `vT[p,q]`$= (\Lambda)_{p,q} = \kappa_{\lambda_{p,q}}$ for $p \neq q$ and `l2T[p,p] = NA`.
`t2B`	$\{\eta_p\}_{p=1}^{P}$	Current MCMC iteration of the global shrinking parameters for the Horseshoe structure of the Voxel coefficients. It is a vector of size P. `t2B[p]`$= \eta_p$.
`l2B`	$\{\phi_{v,p}\}_{v=1,p=1}^{V_p,P}$	Current MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a matrix of size $\times V \times P$. `l2B[v,p]`$= (\Phi)_{v,p} = \phi_{v,p}$ for $v \leq V_p$ and `l2B[v,p]=NA` for $v > V_p$.
`xiB`	κ_{η_p}	Current MCMC iteration of the auxiliary variables of the global shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a vector of size P. `xiT[p]`$= \kappa_{\eta_p}$.
`vB`	$\kappa_{\phi_{v,p}}$	Current MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients .It is a matrix of size $\times V \times P$. `vB[v,p]`$= \kappa_{\phi_{v,p}}$ for $v \leq V_p$ and `vB[v,p] = NA` for $v > V_p$.

Finally the hyper-parameter values of the hyper-priors are passed through from `bmrr_sampler`.

Code	Model	Description
`a_sT`	a_{τ_θ}	Shape hyper-parameter of the Inverse-gamma prior for τ_θ^2. An scalar. By default, `a_sT = 1`.
`b_sT`	b_{τ_θ}	Scale hyper-parameter of the Inverse-gamma prior for τ_θ^2. An scalar. By default, `b_sT = 1`.
`a_sB`	a_{τ_β}	Shape hyper-parameter of the Inverse-gamma prior for τ_β^2. An scalar. By default, `a_sB = 1`.
`b_sB`	b_{τ_β}	Scale hyper-parameter of the Inverse-gamma prior for τ_β^2. An scalar. By default, `b_sB = 1`.
`a_nu`	a_ν	Shape 1 hyper-parameter of the Beta prior for ν. An scalar. By default, `a_sB = 1`.
`b_nu`	b_ν	Shape 2 hyper-parameter of the Beta prior for ν. An scalar. By default, `b_sB = 1`.

4.2.3 Output

The function returns the update of the state of all the variables in the MCMC.

Code	Model	Description
`g`	ξ	Updated MCMC iteration of the region indicator variables. It is a binary vector of size P. `g[p]` $= \xi_p$.
`nu`	ν	Updated MCMC iteration of the probability parameter of the Bernoulli prior of the indicators ξ_p. It is a scalar. `nu`$= \nu$.
`Theta`	Θ	Updated MCMC iteration of the Network coefficients. It is a matrix of size $(\times P \times P)$. `Theta[p,q]`$= (\Theta)_{p,q} = \theta_{p,q}$ for $p \neq q$ and `Theta[p,p]=NA`.
`B`	$\{g_p\}_{p=1}^{P}$	Updated MCMC iteration of the Voxel coefficients. It is a matrix of size $(\times V \times P)$, where $V = \max\{V_p\}_{p=1}^{P}$. `B[v,p]`$= \beta_{v,p}$ for $v \leq V_p$, and `B[v,p]=NA` for $v > V_p$.

`DA`	$\{\psi^a_{p,h}\}_{p=1,h=1}^{P,H}$	Updated MCMC iteration of the bilinear structure of the Network equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^a_{p,h}$.
`DG`	$\{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$	Updated MCMC iteration of the structure of the Voxel equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^g_{p,h}$.
`sT2`	τ^2_θ	Updated MCMC iteration of the variance of the error of the Network equation. It is a scalar. `sT2[s]`$= \tau^2_\theta$.
`sB2`	τ^2_β	Updated MCMC iteration of the variance of the error of the Voxel equation. It is a scalar. `sB2[s]`$= \tau^2_\beta$.
`t2T`	σ^2_θ	Updated MCMC iteration of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `t2T`$= \sigma^2_\theta$.
`l2T`	Λ	Updated MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Network coefficients. It is a
`xiT`	$\kappa_{\sigma^2_\theta}$	Updated MCMC iteration of the auxiliary variable of the global shrinking parameter for the Horseshoe structure of the Network coefficients. It is a scalar. `xiT`$= \kappa_{\sigma^2_\theta}$.
`vT`	$\kappa_{\lambda_{p,p'}}$	Updated MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Network $p \neq q$ and `l2T[p,p] = NA`.
`t2B`	$\{\eta_p\}_{p=1}^{P}$	Updated MCMC iteration of the global shrinking parameters for the Horseshoe structure of the Voxel coefficients. It is a
`l2B`	$\{\phi_{v,p}\}_{v=1,p=1}^{V_p,P}$	Updated MCMC iteration of the local shrinking parameter for the Horseshoe structure of the Voxel coefficients. It is a matrix
`xiB`	κ_{η_p}	Updated MCMC iteration of the auxiliary variables of the global shrinking parameter for the Horseshoe structure of the Voxel
`vB`	$\kappa_{\phi_{v,p}}$	Updated MCMC iteration of the auxiliary variables of the local shrinking parameter for the Horseshoe structure of the Voxel $v \leq V_p$ and `vB[v,p] = NA` for $v > V_p$.

4.2.4 Walkthrough

4.2.4.1 Problem Dimensions

Obtains the relevant dimensions of the data:

- `N`. Number of samples.
- `mV`. $\max\{V_p\}_{p=1}^{P}$.
- `V`$= (V_1, \dots, V_p)'$.
- `P`. Number of regions.
- `Q`. Number of relevant regions.
- `H`. Number of covariates.

4.2.4.2 Samples DT

Loops through every region p and samples according to Full Conditional for $\psi^a_{p,\cdot}$ as follows:

- Computes W.
- Computes R.
- Computes $W'W$.
- Computes $\hat{\psi}^a_{p,\cdot}$.
- Samples $\psi^a_{p,\cdot}$.

4.2.4.3 Samples DT Loops through every region p and samples according to Full Conditional for $\psi^a_{p,\cdot}$ as follows:

- Computes W.
- Computes R.
- Computes $W'W$.
- Computes $\hat{\psi}^g_{p,\cdot}$.
- Samples $\psi^g_{p,\cdot}$.

4.2.4.4 Creates DT, DB and D Creates auxiliary variables:

- `DT`. An auxiliary variable to compute Θ, $\{\beta_p\}_{p=1}^P$ and ξ. It is an array of size $H \times P \times P$. `DT[h,q,p]`$=$ $\psi^a_{q,h}\psi^a_{p,h}$ for $p \neq q$ and `DT[h,q,p]=NA` elsewhere.
- `DB`. An auxiliary variable to compute Θ, $\{\beta_p\}_{p=1}^P$ and ξ. It is an array of size $H \times V \times P$. `DB[h,v,p]`$=$ $\psi^g_{p,h}$ for $v, \leq V_p$ and `DB[h,v,p]=NA` elsewhere.
- `D`. An auxiliary variable to compute τ^2_θ and τ^2_β. It is a matrix of size $H \times P(P-1)/2 + \sum_{p=1}^P V_p$.

4.2.4.5 Samples g, Theta and B Samples the variables that are the main focus of the analysis. To do so it loops through every p to sample jointly $\theta_{-p,p}$, β_p and bgx. Before looping it computes auxiliary variables and then at every iteration `p` it calls the function `group_iterator`. The auxiliary variables, as described in Full conditional for ξ_p, $\Theta_{-p,p}$ and β_p are as follows:

- `AP`. It is an array of size $n \times P \times P$. `AP[i,p,q]`$= a_{i,p,q} - \sum_{h=1}^H \psi^a_{p,h}\psi^a_{p,h}x_{i,h}$.
- `GP`. It is an array of size $n \times V \times P$. `GP[i,v,q]`$= g_{i,v,q} - \sum_{h=1}^H \psi^g_{p,h}x_{i,h}$.
- `Say`. It is a matrix of size $P \times P$. `Say[p,q]`$= \sum_{i=1}^n R^a_{i,p,q}y_i = \sum_{i=1}^n$`AP[i,p,q]`$y_i$.
- `Sgy`. It is a matrix of size $V \times P$. `Sgy[v,p]`$= \sum_{i=1}^n R^g_{i,v,p}y_i = \sum_{i=1}^n$`GP[i,v,p]`$y_i$.
- `Syy`. The sum of squares of y. `Syy`$= \sum_{i=1}^n y_i^2$.

Since there are 2 cases to update jointly $\theta_{-p,p}$, β_p and bgx, that is when $\xi_q = 0$ for every $q \neq p$, in which case only β_p gets updated. If $\xi_q = 1$ at least for q, then $\theta_{-p,p}$ also gets updated, but mot all the entries, only $\theta_{-p,p}[\xi_{-p} = 1]$. Since the procedure samples jointly $\theta_{-p,p}[\xi_{-p} = 1]$ and β_p, and it is returned as only one object by `group_iterator`, then at the end of the loop the coefficients are distributed accordingly.

4.2.4.6 Updates nu Samples ν according to Full conditional ν.

4.2.4.7 Horseshoe Structure for Theta An auxiliary variable is created `gg` that indicates which Network coefficients are non-zero.

Next the horseshoe structure for the Network coefficients is sampled.

4.2.4.7.1 # Samples l2T Samples $\lambda^2_{p,p'}$ according to (9).

4.2.4.7.2 Samples t2T Samples σ^2_θ according to (10).

4.2.4.7.3 Samples vT Samples $\kappa_{\lambda_{p,p'}}$ according to (11).

4.2.4.7.4 Samples xiT Samples $\kappa_{\sigma^2_\theta}$ according to (12).

4.2.4.8 Horseshoe Structure for B An auxiliary variable is created `gV` that indicates which Voxel coefficients are non-zero.

Next the horseshoe structure for the Voxel coefficients is sampled.

4.2.4.8.1 Samples l2B Samples $\phi^2_{v,p}$ according to (14).

4.2.4.8.2 Samples t2B Samples η^2_p according to (15).

4.2.4.8.3 Samples vB Samples $\kappa_{\phi_{v,p}}$ according to (16).

4.2.4.8.4 Samples xiB Samples κ_{η_p} according to (17).

4.2.4.9 Samples sT2 and sB2 In order to sample τ^2_θ and τ^2_β the following auxiliary variables are built:

- `Q`. Number of active regions.
- `Beta`. Network and Voxel coefficients together. `Beta`$= (\theta_{1,2}, \theta_{1,3}, \theta_{2,3}, \dots, \theta_{1,P}, \dots, \theta_{P-1,P}, \beta'_1, \dots, \beta'_P)'$.
- `R`. Matrix of residuals. Each row corresponds to an observation and each column corresponds to a coefficient in the same order as `Beta`.

4.2.4.10 Samples sT2 Samples τ^2_θ as follows:

- Computes $\hat{b}_{\tau_\theta}$.
- Computes $\hat{a}_{\tau_\theta}$.
- Samples τ^2_θ.

4.2.4.11 Samples sB2 Samples τ^2_β as follows:

- Computes $\hat{b}_{\tau_\beta}$.
- Computes $\hat{a}_{\tau_\beta}$.
- Samples τ^2_β.

4.3 `group_iterator`

4.3.1 Description

This function performs the joint sampling of $\theta_{-p,p}$, β_p and ξ_p for a particular p. It follows Full conditional for ξ_p, $\Theta_{-p,p}$ and β_p.

4.3.2 Inputs

This function receives as inputs the variables `Syy`, `Say` and `Sgy` that can be thought as conditional sufficient statistics of the data (conditional on knowing $\{\psi_p^a, \psi_p^g\}_{p=1}^P$) and play a similar role as the sufficient statistics of a regular regression model. It also requires the variance of the errors τ_θ^2 and τ_β^2 and the priors variance to sample $(\theta_{-p,p}, \beta_p)$ jointly according to ξ. To sample ξ_p the prior probability ν is also required Finally, the auxiliary `C` is used and `p` indicates which region is begin sampled.

The data inputs are as follows:

Code	Model	Description
`Say`	S_{ay}	"Conditional sufficient statistic" for the Network Coefficients. It is a matrix of size $P \times P$. `Say[p,q]` $= \sum_{i=1}^n (a_{i,p,q} - \sum_{h=1}^H \psi_{p,h}^a \psi_{p,h}^a x_{i,h}) y_i$
`Sgy`	S_{gy}	"Conditional sufficient statistic" for the Voxel Coefficients. It is a matrix of size $V \times P$. `Sgy[p,q]` $= \sum_{i=1}^n (g_{i,p,q} - \sum_{h=1}^H \psi_{p,h}^g x_{i,h}) y_i$
`Syy`	S_{yy}	The sum of squares of y. `Syy`$= \sum_{i=1}^n y_i^2$.

The other input variables are:

Code	Model	Description
`sT2`	$\tau^2\theta$	Variance of the Network equation error.
`sB2`	$\tau^2\beta$	Variance of the Voxel equation error.
`LT`	$\sigma_\theta^2\Lambda$	Product of the shrinking parameters of the Network Horseshoe prior.
`LB`	$\eta_\beta^2\Phi$	Product of the shrinking parameters of the Voxel Horseshoe prior.
`g`	bgx	Region indicator variables.
`v`	ν	Prior probability for the region indicators.

The set-up inputs are as follows:

Code	Description
`p`	Selected region.
`C`	Is a Boolean matrix of size $V \times P$ that indicates which entries of the voxel data are valid entries. For every observation `i`, `G[i,,][C]`$= g_{i,1}', \dots, g_{i,P}'$.

4.3.3 Output

The outputs of the function are the samples for $\theta_{-p,p}$ and β_p in a single object and the updated version of ξ. Also the probability of a region been selected is returned for analysis purposes.

Code	Model	Description
`b`	$\theta_{-p,p}$ and β_p	This object returns the sample of $\theta_{-p,p}$ and β_p. However, it only returns the samples for the Network coefficients for which $\xi_q = 1$. That is `b`$= (\theta_{-p,p}[\xi_{-p} = 1]', \beta_p')'$.
`g`	ξ	ξ after updating entry p.
`pr`		Full conditional probability of $\xi_p = 1$. For analysis purposes only.

4.3.4 Walktrhough

4.3.4.1 Checks if there are elements in the Symmetric Network If $\xi_q = 0$ for $q \neq p$, then $\theta_{q,p}$ is set as 0 for every q and only β_p is sampled. This section checks that condition and makes the necessary adjustments to compute the auxiliary variables described in Full conditional for ξ_p, $\Theta_{-p,p}$ and β_p.

4.3.4.2 Computes the Log-Odds Computes the Log-odd probability of a region being selected. Notice that after making the necessary adjustments in the previous section two cases are not required. In this way the auxiliary variables c_i are equivalent to `lo` in both cases of Full conditional for ξ_p, $\Theta_{-p,p}$ and β_p.

4.3.4.3 Computes the Probability Computes the probability of the full conditional of ξ_p. It also checks for numerical problems if the odds are equal to `Inf`.

4.3.4.4 Updates g Updates the entry p of ξ.

4.3.4.5 Updates Theta and B Updates $\theta_{-p,p}$ and β_p depending on ξ_p.

4.3.4.6 Returns values Returns the outputs.

4.4 `bmrr_data_sim`

4.4.1 Description

Function to simulate data that can be used on the BMRR model by the [bmrr::bmrr_sampler] function. The data simulated follows the simulation structure of the paper "Multi-Object Data Integration in the Study of Primary Progressive Aphasia." The function includes defaults for every input, according to the "Small dimensional example with high sparsity" setting described in the paper. The data simulation follows the next procedure:

1. Set the dimensions of he data set:

- P The number of regions.
- V The number of voxels per region.
- H The number of additional covariates.
- n The number of observations.

2. Generate indicator variables $\{\xi_p\}_{p=1}^{P}$ for every region that indicate the influential regions. $\xi_p \overset{i.i.d}{\sim} Bernoulli(\nu)$.
3. Simulate the true Voxel Coefficients. For each region p:

- If $\xi_p = 0$, then set all the voxel coefficients in that region to zero. $\beta_p = 0_{V_p}$.
- If $\xi_p = 1$, then simulate a proportion υ_p of the voxels in the region from: $\beta_{v,p} \overset{i.i.d.}{\sim} N(\mu_\beta, \sigma^2_\beta)$. and set the rest of the voxels to 0. Where $\mu_\beta \sim U(l_\beta, u_\beta)$.

4. Simulate the true Network coefficients. For network coefficient $\theta_{p,p'}$ sample:

- If $\xi_p = 1$ and $\xi_{p'} = 1$ simulate from: $\theta_{p,p'} \overset{i.i.d.}{\sim} N(\mu_\theta, \sigma^2_\theta)$. Where $\mu_\theta \sim U(l_\theta, u_\theta)$.
- If $\xi_p = 0$ or $\xi_{p'} = 0$ set $\theta_{p,p'} = 0$.

5. Simulate the bilinear coefficient structure of the Network equation. $\psi^a_{p,h} \overset{i.i.d.}{\sim} N(\mu_{\psi^a}, \sigma^2_{\psi^a})$ for $p = 1, \dots, P$ and $h = 1, \dots, H$.
6. Simulate the covariate coefficients of the Voxel equation. $\psi^g_{p,h} \overset{i.i.d.}{\sim} N(\mu_{\psi^g}, \sigma^2_{\psi^g})$ for $p = 1, \dots, P$ and $h = 1, \dots, H$.

7. Simulate the covariate of interest. $y_i \overset{i.i.d.}{\sim} N(0, \sigma^2_y)$ for $i = 1, \dots, n$.
8. Simulate the additional covariates. For each covariate h:
    - If the covariate is desired to be an indicator simulate from: $x_{i,h} \overset{i.i.d.}{\sim} Bernoulli(\nu_{x_h})$ for $i = 1, \dots, n$.
    - If the covariate is continuous simulate from: $x_{i,h} \overset{i.i.d.}{\sim} N(0, \sigma^2_x)$ for $i = 1, \dots, n$.
9. Simulate a Network object. $a_{i,(p,p)} = \sum_{h=1}^{H} \psi^a_{p,h} \psi^a_{p',h} x_{i,h} + \theta_{p,p'} y_i + e_i^{(p,p')} \quad {}^{p<p'}$ where $e_i^{p,p'} \overset{i.i.d.}{\sim} N(0, \tau^2_\theta)$ for $i = 1, \dots, n$. The same as the Network equation in (1).
10. Simulate a Voxel object. $g_{i,(v,p)} = \sum_{h=1}^{H} \psi^g_{p,h} x_{i,h} + \beta_{v,p} y_i + w_i^{(p,p')} \quad p \in 1 < \dots, P, \quad v \in 1, \dots, V_p$ where $w_i^{v,p} \overset{i.i.d.}{\sim} N(0, \tau^2_\beta)$ for $i = 1, \dots, n$. The same as the Network equation in (2).

4.4.2 Inputs

The function receives as inputs the dimension parameters and the simulation parameters for the variables. The dimension parameters are:

Code	Model	Description
P	P	Number of regions.
V	V	A vector of size P, with he number of voxels per region.
N	n	Number of observations.
H	H	Number of additional covariates.

The simulation parameters are:

Code	Model	Description
nu	ν	Probability of a region being selected as influential.
u	υ	Proportion of non-zero voxels per influential region.
cB	l_β, u_β	Minimum and maximum of the Uniform distribution from which the mean of the voxel coefficients is simulated.
cT	l_θ, u_θ	Minimum and maximum of the Uniform distribution from which the mean of the Network coefficients is simulated.
s2B	σ^2_β	Variance of the Normal from which the voxel coefficients are simulated.
s2T	σ^2_θ	Variance of the Normal from which the Network coefficients are simulated.
cDA	μ_{ψ^a}	Mean of the bilinear Network structure variables.
cDG	μ_{ψ^g}	Mean of the additional covariates voxel coefficients.
s2DA	$\sigma^2_{\psi^a}$	Variance of the bilinear Network structure variables.
s2DG	$\sigma^2_{\psi^g}$	Variance of the additional covariates voxel coefficients.
s2A	σ^2_θ	Variance of error of the Network equations.
s2G	σ^2_β	Variance of error of the Voxel equations.

Finally, an input for the type of additional covariates.

Code	Description
p=covInd	A Boolean vector of size H indicating if the covariate is an indicator variable (TRUE) or not (FALSE).

4.4.3 Output

The function returns a data set for which BMRR can be used, as well as the parameters from which it was simulated, that can be used to measure the performance of BMRR.

The data set output is as follows:

Code	Model	Description
`y`	y	The variable of interest. It is a vector of size `N`.
`A`	A	The Network object. It is an array of size $(n \times P \times P)$.
`G`	$\{g_p\}_{p=1}^{P}$	The voxel object. It is an array of size $(n \times V \times P)$, where $V = \max(\{V_p\}_{p=1}^{P})$. If the regions are not the same size (i.e. there is a p for which $V_p < V$), then the entry `G[i,p,]` is padded with `NA`.
`X`	X	Additional covariates. It is a matrix of size $(n \times H)$. Each column `X[,h]` represents one covariate.

The data generating parameters are:

Code	Model	Description
`g`	ξ	The region indicator variables. It is a binary vector of size P. `g[p]` $= \xi_p$.
`Theta`	Θ	The Network coefficients. It is a matrix of size $(\times P \times P)$. `Theta[p,q]`$= (\Theta)_{p,q} = \theta_{p,q}$ for $p \neq q$ and `Theta[p,p]=NA`.
`B`	$\{g_p\}_{p=1}^{P}$	The Voxel coefficients. It is a matrix of size $(\times V \times P)$, where $V = \max\{V_p\}_{p=1}^{P}$. `B[v,p]`$= \beta_{v,p}$ for $v \leq V_p$, and `B[v,p]=NA` for $v > V_p$.
`DA`	$\{\psi^{a}_{p,h}\}_{p=1,h=1}^{P,H}$	The bilinear structure of the Network equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^{a}_{p,h}$.
`DG`	$\{\psi^{g}_{p,h}\}_{p=1,h=1}^{P,H}$	The structure of the Voxel equation coefficients of the covariates X. It is a matrix of size $(\times P \times H)$. `DA[p,h]`$= \psi^{g}_{p,h}$.

4.4.4 Walkthrough

4.4.4.1 Maximum Number of Voxels Obtains the maximum number of voxels per region. This variable determines the size of `G` and `B`.

4.4.4.2 Computes Active Regions Selects the influential regions. `g`ξ. Step 2 of Description of `bmrr_data_sim`.

4.4.4.3 Computes Active Voxels Creates a variable `gB` that indicates which voxels are non-zero.

4.4.4.4 Creates B The Voxels coefficients of the variable of interest. `B`$= \{g_p\}_{p=1}^{P}\$$. Step 3 of Description of `bmrr_data_sim`.

4.4.4.5 Creates Theta The Network coefficients of the variable of interest. `Theta`$= \Theta$. Step 4 of Description of `bmrr_data_sim`.

4.4.4.6 Creates DA The bilinear structure of the coefficients of the additional covariates of the Network equations. `DA`$= \{\psi^{a}_{p,h}\}_{p=1,h=1}^{P,H}$. Step 5 of Description of `bmrr_data_sim`.

4.4.4.7 Creates DG The structure of the coefficients of the additional covariates in the Voxel equations. `DA`$= \{\psi^g_{p,h}\}_{p=1,h=1}^{P,H}$. Step 6 of Description of `bmrr_data_sim`.

4.4.4.8 Samples y Simulates the variable of interest. `y`$= y$. It is standardized. Step 7 of Description of `bmrr_data_sim`.

4.4.4.9 Samples X Simulates the additional covariates, according to the indicator `covInd`. If, applicable it is standardized. `X`$= X$. Step 8 of Description of `bmrr_data_sim`.

4.4.4.10 Samples A Samples the Network object. `A`$= A$. Step 9 of Description of `bmrr_data_sim`.

4.4.4.11 Samples G Samples the Voxel object. `G`$\{g_p\}_{p=1}^{P}$. Step 10 of Description of `bmrr_data_sim`.

4.4.4.12 Centers Centers the objects `A` and `G` by edge and voxel respectively.

4.4.4.13 Returns Values Returns the data set and the simulation variables.

5 Example

In this chapter the use of [bmrr::bmrr_sampler] is illustrated with data generated by [bmrr::bmrr_data_sim]. The example uses the generating parameters of the paper “Multi-Object Data Integration in the Study of Primary Progressive Aphasia” for the “Small Dimensional” case with a sparsity of 85% and 70%.

If the package is not installed, enter:

```
library(devtools)
install_github("Rene-Gutierrez/bmrr")
```

Then load the package:

```r
library(bmrr)
```

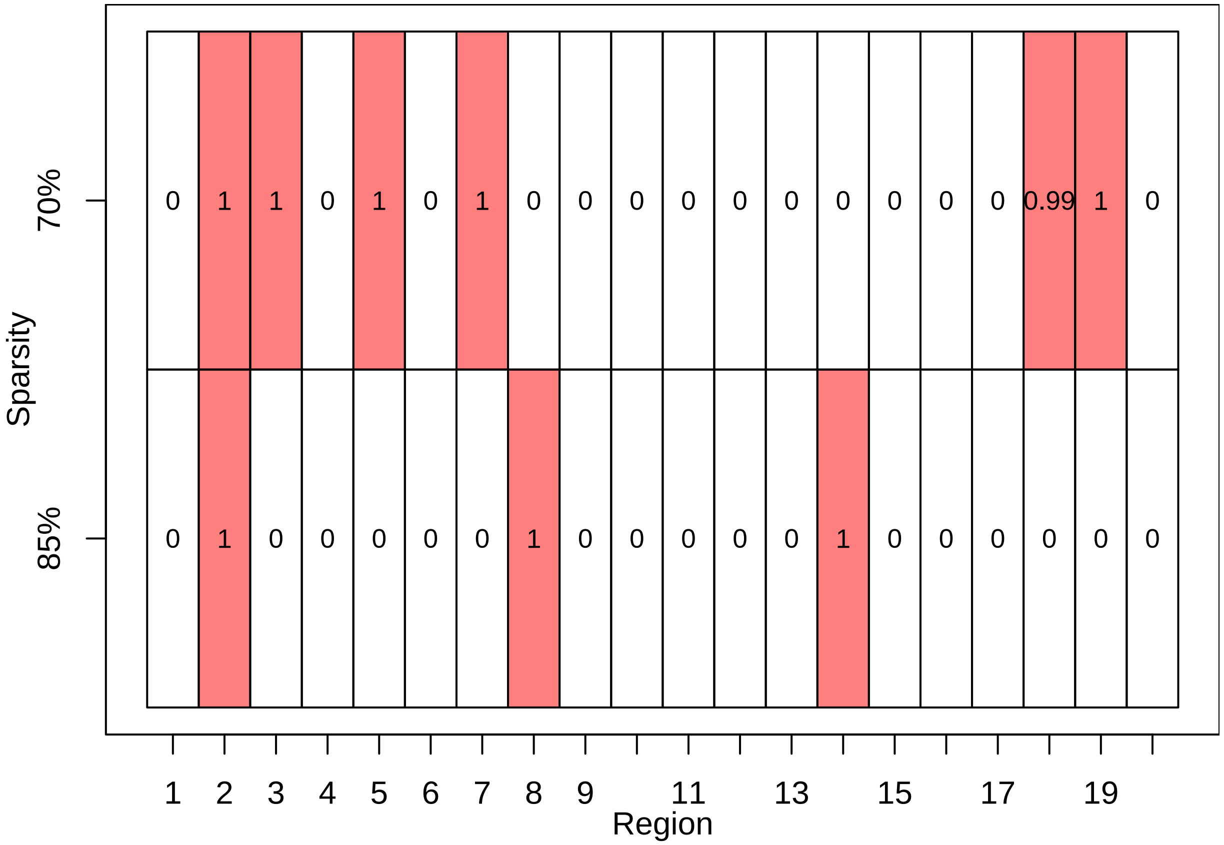
The generate a data set with `bmrr_data_sim` and then using the generated data set a run of `bmrr_sampler`.

```r
set.seed(02092023)
dat_1 <- bmrr_data_sim(P      = 20,
                       V      = rep(10, 20),
                       N      = 25,
                       H      = 3,
                       nu     = 0.15,
                       u      = rep(0.4, 20),
                       cB     = c(0.25, 1),
                       cT     = c(0.25, 1),
                       s2B    = 1,
                       s2T    = 1,
                       cDA    = 0,
                       cDG    = 0,
                       s2DA   = 1,
                       s2DG   = 1,
                       s2A    = 1,
                       s2G    = 1)

dat_2 <- bmrr_data_sim(P      = 20,
                       V      = rep(10, 20),
                       N      = 25,
                       H      = 3,
                       nu     = 0.30,
                       u      = rep(0.4, 20),
                       cB     = c(0.25, 1),
                       cT     = c(0.25, 1),
                       s2B    = 1,
                       s2T    = 1,
                       cDA    = 0,
                       cDG    = 0,
                       s2DA   = 1,
                       s2DG   = 1,
                       s2A    = 1,
                       s2G    = 1)

out_1 <- bmrr_sampler(y          = dat_1$y,
                      X          = dat_1$X,
                      G          = dat_1$G,
                      A          = dat_1$A,
                      nmcmc      = 5000,
                      burnin     = 1000,
                      thinning   = 1)

out_2 <- bmrr_sampler(y          = dat_2$y,
                      X          = dat_2$X,
                      G          = dat_2$G,
                      A          = dat_2$A,
                      nmcmc      = 5000,
                      burnin     = 1000,
                      thinning   = 1)
```

Now a plot like Figure 2 of the paper is generated.

```
xmin <- 0
xmax <- 20
ymin <- 0
ymax <- 2
P    <- 20
par(mar = c(3.5, 3.5, 0, 0))
plot(-1,
     type = 'l',
     xlab = "",
     ylab = "",
     ylim = c(ymin, ymax),
     xlim = c(xmin, xmax),
     yaxt = 'n',
     xaxt = 'n',
```

```r
     lwd  = 2)
rect(xleft   = (1:P)[dat_1$g == 1] - 1,
     xright  = (1:P)[dat_1$g == 1],
     ybottom = rep(1, P)[dat_1$g == 1] - 1,
     ytop    = rep(1, P)[dat_1$g == 1],
     density = NA,
     col     = rgb(1, 0, 0, 0.5),
     border  = "black")
rect(xleft   = (1:P)[dat_1$g == 0] - 1,
     xright  = (1:P)[dat_1$g == 0],
     ybottom = rep(1, P)[dat_1$g == 0] - 1,
     ytop    = rep(1, P)[dat_1$g == 0],
     density = NA,
     col     = rgb(1, 1, 1, 0.5),
     border  = "black")
text(x       = 1:P - 0.5,
     y       = rep(0, P) + 0.5,
     labels  = round(colMeans(out_1$sam$g), 2),
     cex     = 0.85)
rect(xleft   = (1:P)[dat_2$g == 1] - 1,
     xright  = (1:P)[dat_2$g == 1],
     ybottom = rep(1, P)[dat_2$g == 1],
     ytop    = rep(1, P)[dat_2$g == 1] + 1,
     density = NA,
     col     = rgb(1, 0, 0, 0.5),
     border  = "black")
rect(xleft   = (1:P)[dat_2$g == 0] - 1,
     xright  = (1:P)[dat_2$g == 0],
     ybottom = rep(1, P)[dat_2$g == 0],
     ytop    = rep(1, P)[dat_2$g == 0] + 1,
     density = NA,
     col     = rgb(1, 1, 1, 0.5),
     border  = "black")
text(x       = 1:P - 0.5,
     y       = rep(0, P) + 1.5,
     labels  = round(colMeans(out_2$sam$g), 2),
     cex     = 0.85)
axis(side = 1, at = 0:(P - 1) + 0.5, labels = 1:P)
axis(side = 2, at = 1:2 - 0.5,        labels = c("85%", "70%"))
title(ylab = "Sparsity",
      line = 1.8,
      xlab = "Region")
```



This file is available to edit as 04-example at Github.

References

Carlos M Carvalho, Nicholas G Polson, and James G Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

Alan Genz and Frank Bretz. *Computation of Multivariate Normal and t Probabilities*. Lecture Notes in Statistics. Springer-Verlag, Heidelberg, 2009. ISBN 978-3-642-01688-2.

Alan Genz, Frank Bretz, Tetsuhisa Miwa, Xuefei Mi, Friedrich Leisch, Fabian Scheipl, and Torsten Hothorn. *mvtnorm: Multivariate Normal and t Distributions*, 2021. URL https://CRAN.R-project.org/package=mvtnorm. R package version 1.1-3.

Edward I George and Robert E McCulloch. Approaches for Bayesian variable selection. *Statistica sinica*, 7: 339–373, 1997.

Enes Makalic and Daniel F. Schmidt. A Simple Sampler for the Horseshoe Estimator. *IEEE Signal Processing Letters*, 23(1), January 2016. ISSN 1558-2361. doi: 10.1109/LSP.2015.2503725.