Journal of Statistical Software

August 2024, Volume 110, Issue 3. *doi: 10.18637/jss.v110.i03*

makemyprior: Intuitive Construction of Joint Priors for Variance Parameters in R

Ingeborg Hem
Norwegian University of Science and Technology

Geir-Arne Fuglstad
Norwegian University of Science and Technology

Andrea Riebler
Norwegian University of Science and Technology

Abstract

Priors allow us to robustify inference and to incorporate expert knowledge in Bayesian hierarchical models. This is particularly important when there are random effects that are hard to identify based on observed data. The challenge lies in understanding and controlling the joint influence of the priors for the variance parameters, and **makemyprior** is an R package that guides the formulation of joint prior distributions for variances. A joint prior distribution is constructed based on a hierarchical decomposition of the total variance in the model along a tree, and takes the entire model structure into account. Users input their prior beliefs or express ignorance at each level of the tree. Prior beliefs can be general ideas about reasonable ranges of variance values and need not be detailed expert knowledge. The constructed priors lead to robust inference and guarantee proper posteriors. A graphical user interface facilitates construction and assessment of different choices of priors through visualization of the tree and joint prior. The package aims to expand the toolbox of applied researchers and make priors an active component in their Bayesian workflow.

Keywords: Bayesian hierarchical models, robust inference, joint prior distributions, hierarchical variance decomposition, graphical user interface, R.

1. Introduction

Bayesian modelling is more available than ever through fast and easy-to-use software programs for Bayesian inference such as integrated nested Laplace approximations (INLA, Rue, Martino, and Chopin 2009) in R (R Core Team 2024) through **INLA** (see `https://www.R-INLA.org/`), Stan (Carpenter *et al.* 2017) with the R interface **rstan** (Stan Development Team 2024) and dependencies such as **rstanarm** (Goodrich, Gabry, Ali, and Brilleman 2020), **brms** (Bürkner 2017), **shinystan** (Gabry and Veen 2022) and **loo** (Vehtari *et al.* 2020), **WinBUGS** (Lunn,

Thomas, Best, and Spiegelhalter 2000) with the R interface **R2WinBUGS** (Sturtz, Ligges, and Gelman 2005), **OpenBUGS** (Spiegelhalter, Thomas, Best, and Lunn 2014) with the R interface **R2OpenBUGS** (Gelman, Sturtz, Ligges, Gorjanc, Kerman, and Thomas 2020a), Template Model Builder (**TMB**, Kristensen, Nielsen, Berg, Skaug, and Bell 2016), **JAGS** (Plummer 2017), Bayesian analysis toolkit (**BAT**, Caldwell, Kollár, and Kröninger 2009), and more. These programs offer many ways to construct complex models suitable for a wide range of applications. Default settings for priors and hyperparameters are usually given, and instructions on how to change the default settings can be found in the software documentation. However, even though there is an increasing focus on the fact that prior distributions should be chosen consciously (Zondervan-Zwijnenburg, Peeters, Depaoli, and Van de Schoot 2017; Gelman *et al.* 2020b; Smid and Winter 2020), guides on how the priors should be chosen are missing. Our goal with **makemyprior** is to close this gap and to increase the awareness of prior choices. The package is available from the Comprehensive R Archive Network (CRAN): `https://CRAN.R-project.org/package=makemyprior`.

A flexible and much used class of models where Bayesian modeling can be employed is Bayesian hierarchical models (e.g., Gelman and Hill 2007; Cressie and Wikle 2011; Gelman, Carlin, Stern, Dunson, Vehtari, and Rubin 2013; Banerjee, Carlin, and Gelfand 2014). In Bayesian hierarchical models, the variation in the observations is modeled through a combination of an observation model, a latent model, and priors for the parameters. The observation model defines a generative model for the observed data conditional on the latent model. The link between the latent model and each observation acts through the transformation of a linear predictor, which we assume is a linear combination of fixed and random effects. The goal of the linear predictor is to explain the variation in the true signal. We concentrate on latent Gaussian models, where the linear predictor is composed of random model components that follow multivariate Gaussian distributions conditional on the parameters. These parameters control the random effects and need prior distributions. We focus on the most central type of model parameters: the variance parameters.

Parameters controlling means and medians, such as the coefficients of fixed effects, are close to the data and tolerate vague priors (Goel and Degroot 1981; Gelman *et al.* 2020b). The specification of a prior distribution for a variance parameter is challenging (Lambert, Sutton, Burton, Abrams, and Jones 2005; Gelman, Simpson, and Betancourt 2017), but is at the same time a strong feature of Bayesian inference. Prior knowledge, obtained from previous experiments or comparable investigations, and expert knowledge can be included to make the model more robust. It is convenient to use the default priors included in programs for inference, however, by using defaults we are not fully utilizing the Bayesian framework. Currently, there are no other R packages that intuitively allows the proper inclusion of such knowledge and simple visualization of chosen priors in a straightforward manner.

With **makemyprior** we empower users to actively select priors that are suitable for their model structure and application at hand. We guide the user to formulate sensible prior distributions, that are automatically computed and visualized in an intuitive manner, and the user is made aware of which priors are used in the model and what they express. In particular, the assumptions underlying the default settings are made clear. Package **makemyprior** does this by applying the hierarchical decomposition (HD) prior framework proposed by Fuglstad, Hem, Knight, Rue, and Riebler (2020), where the variation in the observed data is distributed to the random effects following a *prior tree structure*. In this tree, the leaf nodes represent the random effects specified in the linear predictor, and the top (root) node represents the sum

of the random effect variance, denoted the *total variance*. How much variance is distributed from a parent node to its child nodes is determined by a split in the tree, and this procedure continues down until the leaf nodes. This gives us a parameterization with *proportions of variance*, instead of the more common variance parameter parameterization. The priors for those variance proportion parameters can be specified intuitively and transparently as they often coincide with the scale on which prior or expert knowledge exists, such as in genomic modeling (Holand, Steinsland, Martino, and Jensen 2013; Hem, Selle, Gorjanc, Fuglstad, and Riebler 2021) and disease mapping (Wakefield 2006). Fixed effects are kept out of the hierarchical prior and given Gaussian priors with zero mean and a fixed high variance.

Prior distributions can be either specified directly within R or through a graphical user interface (GUI). In the GUI the user can inspect the prior tree and adapt it as needed. One can click through the splits independently to specify the beliefs for each split. A guide asking relevant questions about the model to ease the prior specification process further is also available. The user can be ignorant and distribute the variance equally to the child nodes through a Dirichlet prior, or exploit expert knowledge implemented via penalized complexity (PC, Simpson, Rue, Riebler, Martins, and Sørbye 2017) priors. After completing the prior specification, inference can be carried out directly with the R packages **rstan** and **INLA**.

We begin with explaining the concepts of total variance and hierarchical variance decomposition through two motivating examples in Section 2. We then introduce the necessary background in Section 3 before we give an overview of the software in Section 4 and then present the **makemyprior** package with general explanations on how to use it in Section 5. Section 6 gives more detailed examples showing how to use the package in various situations. A summary and discussion is given in Section 7.

2. Motivating examples

We demonstrate the core ideas of total variance and hierarchical decomposition of the total variance through two illustrative examples. These examples are simplified versions of examples we use in Section 6.

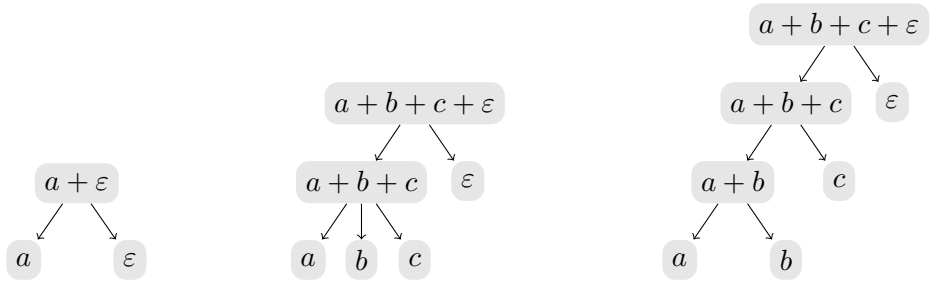
One of the key quantities in quantitative genetics concerns the distribution of observed variation to genetic and environmental sources. In this setting, good intuition exists on the ratio of genetic to phenotypic variation, also known as the *heritability*, whereas it is more difficult to define suitable priors separately on the two variance parameters.

Example 2.1 (Genomic models)**.** Consider a group of n individuals, where each individual i has an observed phenotype y_i. A simple genomic model is:

$$y_i = \mu + a_i + \varepsilon_i, \quad i = 1, \ldots, n,$$

where μ is an intercept, $\boldsymbol{a} = (a_1, \ldots, a_n)^\top \sim \mathcal{N}_n(\mathbf{0}, \sigma_\text{a}^2 \mathbf{A})$ is an additive genetic effect, and $\boldsymbol{\varepsilon} = (\varepsilon_1, \ldots, \varepsilon_n)^\top \sim \mathcal{N}_n(0, \sigma_\varepsilon^2 \mathbf{I}_n)$ is environmental noise. The covariance matrix $\mathbf{A}$ is calculated based on genetic sequencing of the n individuals and is scaled so that σ_a^2 is representative of the variance arising from the genetic effect, see Selle, Steinsland, Hickey, and Gorjanc (2019); Hem *et al.* (2021) for details.

In this simple model, two key quantities are the total variance $\sigma_\text{P}^2 = \sigma_\text{a}^2 + \sigma_\varepsilon^2$, also known as the *phenotypic variance*, and the heritability $h^2 = \sigma_\text{a}^2/(\sigma_\text{a}^2 + \sigma_\varepsilon^2)$, which is the proportion of



(a) Genomic model. (b) Latin square model. (c) Latin square model.

Figure 1: Prior trees for (a) the genomic model in Example 2.1, and (b, c) the latin square model in Example 2.2.

the phenotypic variance explained by the genetic effect. When expert knowledge is available about these two quantities, this information can be directly exploited through a joint prior assigned to phenotypic variance and heritability. A simple and intuitive visualization of this parameterization is given by the tree in Figure 1a where the phenotypic variance $\sigma^2_{\text{P}} = \sigma^2_{\text{a}} + \sigma^2_{\varepsilon}$ in the top (root) node is distributed to the additive genetic variance σ^2_{a} and the environmental variance σ^2_{ε} in the two leaf nodes. Note that the intercept is treated independently with a vague Gaussian prior. See Hem *et al.* (2021) for a detailed description.

The idea of expressing a parameterization that is given in terms of total variance and proportions of variances through a tree, extends to more complex models with more random effects. For example, when analyzing data arising from designed experiments.

Example 2.2 (Latin square design)**.** Based on the ideas in Fuglstad *et al.* (2020), we assume that an agricultural field is split into rows and columns resulting in a 9×9 grid, and that one out of 9 different strengths of fertilizer is applied in each grid cell. Outcomes $y_{i,j}$ are observed in row i and column j under treatment $k[i,j]$, and modeled through

$$y_{i,j} = \alpha + \beta \cdot k[i,j] + a_i + b_j + c_{k[i,j]} + \varepsilon_{i,j}, \quad i,j = 1,\dots,9,$$

where α is an intercept and $\beta \cdot k[i,j]$ is a linear effect of treatment. α and β are assigned vague Gaussian priors $\mathcal{N}(0, \sigma = 1000)$. $\boldsymbol{a} = (a_1,\dots,a_9)^\top \sim \mathcal{N}_9(\mathbf{0}, \sigma^2_{\text{a}}\mathbf{I}_9)$ is a row effect, $\boldsymbol{b} = (b_1,\dots,b_9)^\top \sim \mathcal{N}_9(\mathbf{0}, \sigma^2_{\text{b}}\mathbf{I}_9)$ is a column effect, $\boldsymbol{c} = (c_1,\dots,c_9)^\top \sim \mathcal{N}_9(\mathbf{0}, \sigma^2_{\text{c}}\mathbf{I}_9)$ is a treatment effect, and the residual noise is $\boldsymbol{\varepsilon} = (\varepsilon_{1,1}, \varepsilon_{1,2}\dots,\varepsilon_{9,9})^\top \sim \mathcal{N}_{81}(\mathbf{0}, \sigma^2_{\varepsilon}\mathbf{I}_{81})$. $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ have sum-to-zero constrains. The individual variances, σ^2_{a}, σ^2_{b}, σ^2_{c} and σ^2_{ε}, are nuisance parameters, and it may be difficult to have prior knowledge about them.

Figures 1b and 1c visualize two ways in which the total variance $\sigma^2_{\text{a+b+c+}\varepsilon} = \sigma^2_{\text{a}} + \sigma^2_{\text{b}} + \sigma^2_{\text{c}} + \sigma^2_{\varepsilon}$ can be distributed to the individual variances σ^2_{a}, σ^2_{b}, σ^2_{c} and σ^2_{ε} in the leaf nodes. Using Figure 1b, we could envision that, in the top split, shrinkage is applied to the latent variance $\sigma^2_{\text{a+b+c}} = \sigma^2_{\text{a}} + \sigma^2_{\text{b}} + \sigma^2_{\text{c}}$ relative to the residual variance σ^2_{ε} with the goal of reducing the risk of overfitting. Then in the second split, we could envision that we want to express ignorance about how the latent variance is distributed to the individual variances σ^2_{a}, σ^2_{b} and σ^2_{c}. Alternatively, using Figure 1c, we may want to express ignorance about how $\sigma^2_{\text{a+b}} = \sigma^2_{\text{a}} + \sigma^2_{\text{b}}$ is distributed to σ^2_{a} and σ^2_{b}, but apply shrinkage to the variance of the treatment variance σ^2_{c} relative to $\sigma^2_{\text{a+b}}$. The full details can be found in Fuglstad *et al.* (2020).

Examples 2.1 and 2.2 make it clear that in some cases it is natural to think in terms of proportions of variances. However, explicitly writing out how the proportions are defined may obfuscate the key ideas that one want to express. Therefore, trees such as shown in Figure 1 are critical to define such priors.

We want to emphasize that a prior can be chosen in many ways, and there is no one right choice. Our message is that it is a choice to use the default prior, it is a choice to use literature-based priors, and it is a choice to use a prior based on prior and expert knowledge. We believe it is important to communicate this and to make prior selection an active part of the Bayesian workflow.

3. Background

In this section, we present the necessary theoretical and methodological background behind the **makemyprior** package and introduce terminology used throughout the paper.

3.1. Univariate variance priors

The traditional strategy for selecting priors for variance parameters is to choose separate univariate prior distributions for each variance parameter in the model. Any distribution with support for non-negative numbers can be used, both distribution families such as Half-Cauchy and inverse gamma, and distributions derived from principles. Principle-based priors are priors that are based on generic principles independent of model and application, and are not parameter specific, but rather constructed on a general model parameter and transformed to the parameter of interest.

One such prior is the penalized complexity (PC) prior of Simpson *et al.* (2017). They use the *distance* $d(\cdot)$ between a base model and a flexible extension of this base model, measured using the Kullback-Leibler divergence. In the base model the parameter of interest θ is fixed to θ_0. In the flexible model θ is allowed to vary, and the PC prior induces shrinkage towards the base model, which gives a robust prior that aids to avoid overfitting. A good prior distribution can improve the robustness of the inference, by helping to avoid estimating spurious effects. For a random effect with zero mean and variance parameter σ^2, the base model is a model where $\sigma^2 = 0$, meaning the effect does not contribute in the model. This gives a simpler model with one fewer effect. When the parameter is no longer equal to the base model value, we move away from the base model, and this deviation is penalized, ensuring that we do not overfit the model. As Simpson *et al.* (2017), we use an exponential prior for the distance $d(\cdot)$ between the two models, and transform this to a prior for the desired parameter. This means that the PC prior always is an exponential distribution on the distance, while for the parameter in question θ the distribution varies with parameterization, choice of base model and covariance matrix structures. The parameter θ can be for example a standard deviation (Simpson *et al.* 2017), a correlation parameter (Guo, Riebler, and Rue 2017), or a proportion of variances (Fuglstad *et al.* 2020; Hem *et al.* 2021). The PC prior does in general not have an analytical expression when transformed to the parameter of interest. However, for a standard deviation the distance is simply $d(\sigma) = \sigma$ (Simpson *et al.* 2017), meaning the PC prior is the exponential distribution for a standard deviation. The hyperparameters are chosen using prior knowledge on an interpretable scale relevant for the parameter, such as a tail probability for a standard deviation σ, $\mathsf{P}(\sigma > U) = \alpha$.

Other principle-based priors include the reference priors, which originate from the idea that the prior should to as large extent as possible be dominated by the data, and the posterior should be influenced as little as possible by the prior (Berger, Bernardo, and Sun 2009). For a single parameter, the reference prior becomes Jeffreys' prior (Jeffreys 1946). Jeffreys' prior on a variance parameter is improper without support in 0, which is useful when we do not want to say anything about the scale of the variance.

3.2. Hierarchical variance decomposition along a prior tree

For direct control of the shrinkage of the variance parameters of the different model components relative to each other such as in Examples 2.1 and 2.2, it is more natural to work with joint variance priors than univariate variance priors. We describe one way to construct such priors through trees.

General definition of a prior tree

A prior tree is a directed acyclic graph consisting of a *top node*, *split nodes* and *leaf nodes*, connected by directed edges. Each random model component is represented by a leaf node. The top node in the tree represents total variance, i.e., the sum of the variance of the leaf nodes (random effects). We follow Fuglstad *et al.* (2020) and consider variance that is not explained by fixed effects, and therefore omit fixed effects from the hierarchical decomposition (HD) prior and the prior tree. Two or more leaf nodes are combined in a split node in a way that reflects the hierarchical model and our prior beliefs about how the total variance is distributed. The split nodes represent a variance proportion.

In some cases we only want to include a subset of the model components in the same prior tree, and in that way have several trees. An example is the animal model (e.g., Holand *et al.* 2013), a mixed model that is usually used to decompose environmental and genetic variances in animal populations. The genetic contribution might be split into several sub-components, like additive, dominance and mutational variance. This is an extension of the genomic model in Example 2.1. We may have a good intuition on the absolute magnitude of the variation that is explained by the environment and by all the genetic contributions separately, but only have knowledge about the relative magnitude of the genetic variation. The genetic contributions are often confounded, and it is useful to split the variance of the genetic contribution using a prior tree. In that way we have both variance parameters, total variance parameters, and variance proportions.

If a variance component is assigned an individual prior, its corresponding random effect is represented by a *singleton*: A node not connected to any other nodes. The singletons can be considered as trees with only one node. Several prior trees gives a prior forest. We refer to the forest of trees as the *prior tree structure* of the model. Each tree in a tree structure is associated with their own joint prior distribution, independent of the priors belonging to the other trees. The tree structure is made based on prior knowledge about the model, data and problem at hand.

Defining priors for the split nodes: Shrinkage versus ignorance

Given a tree structure describing the distribution of the variance in the model, we can use the rest of our pre-existing knowledge to steer the variance to the different model components by choosing suitable priors for the different parameters belonging to the prior tree.

We apply the PC prior of Simpson *et al.* (2017) as it is robust by design due to its shrinkage properties. Any model parameter can be assigned a PC prior. For a variance proportion parameter, which steers the variance down the prior tree, the base model can be any value in $[0, 1]$. In the corner cases 0 and 1, we are in a similar situation as for the variance parameter, where the base model represents a model with fewer random effects as one of them shrinks away. The heritability h^2 in Example 2.1 is a variance proportion. We can imagine that a geneticist says that the heritability is around 0.4 for this phenotype and species, but that it is unsure whether the additive effect contributes or not. Then we can use a PC prior that shrinks the heritability to 0, which gives only residual effect in the base model, and ensure that the prior has a median of 0.4, i.e., $\mathsf{P}(h^2 > 0.4) = 0.5$.

If the base model is somewhere between 0 and 1, we express with the prior that both effects involved in the variance proportion should be present in the model. If the geneticist is certain that the additive effect is present in the model, we can choose a PC prior that shrinks the heritability h^2 towards 0.4, and with median at the same value: $\mathsf{P}(h^2 > 0.4) = 0.5$. This corresponds to 40% additive and 60% residual effect in the base model. See Section 3.3 and Fuglstad *et al.* (2020) for details.

By using a *multi-split*, a split with more than two child nodes, the user expresses no strong opinions on how the variance is distributed among the components involved in the split. Then we use a prior that assigns an equal amount of variance to each of the model effects involved in the split through the ignorant symmetric Dirichlet prior. This ignorant prior can also be used on a dual split, in which case it reduces to a uniform prior on $[0, 1]$, and with that the user expresses that no prior information is used in that split.

Defining priors for top nodes and singletons

Appropriate prior distributions for variance parameters vary with the likelihood. We describe the train of thought when specifying prior distributions for variance parameters when the likelihood is Gaussian, binomial and Poisson.

In a model with a Gaussian likelihood, the total variance is usually easy to identify and does not need an informative prior. This is a parameter close to the data in the model and settles with a vague prior (Goel and Degroot 1981; Gelman *et al.* 2020b). For the total variance in the top nodes, Fuglstad *et al.* (2020) recommend the scale-invariant, improper Jeffreys' prior when all model components are involved in one single tree. This prior does not require any hyperparameters and is straight-forward to use.

In cases where the user has specific knowledge about the total variance, a proper prior can be used to include this knowledge in the prior. In cases where the nodes are not all involved in the same tree and we have a prior forest, a scale-invariant prior is not meaningful, and an improper prior may lead to an improper posterior. This implies that singletons always must be assigned a proper prior. Due to its desirable shrinking properties, Fuglstad *et al.* (2020) recommend the PC prior (Simpson *et al.* 2017) for the variances. We denote this prior $\text{PC}_0(U, \alpha)$, where the subscript indicates that the shrinkage is towards 0 and U and α are hyperparameters controlling the tail probability of the prior. For the genomic model in Example 2.1, assume we have prior or expert knowledge saying it is unlikely that the total variance in the observed data is greater than 4. We want to use this knowledge, and choose $U = \sqrt{4}$. The value we choose for α says something about how certain we are in the value of U. Using $\alpha = 0.05$ is a suitable choice, so that $\mathsf{P}(\sigma > \sqrt{4}) = 0.05$ and $\sigma \sim \text{PC}_0(\sqrt{4}, 0.05)$.

Other likelihoods require proper priors on all variances, also the total variance, as scale-invariance is not meaningful for data that are not Gaussian. Again we follow the recommendation of Fuglstad *et al.* (2020) and suggest PC priors. However, instead of choosing a prior using the upper tail probability of the standard deviation, we think on a different scale than for Gaussian data, and choose a credible interval for the variance parameter on a suitable scale. Then we transform the interval into an upper tail probability $\mathsf{P}(\sigma > U) = \alpha$. For both binomial likelihood with logit link and Poisson likelihood with log link, an exponential scale is appropriate. This will correspond to thinking on odds-ratio scale for binomial data, and on the scale of the data for Poisson data.

Assume we have a model with linear predictor $\eta_i = \mu + a_i + b_i$ and binomial likelihood with logit link function. An intuitive way of choosing a prior for the total variance of $a_i + b_i$ is to choose an equal-tailed credible interval for the effect of the random effects on the odds-ratio, $\exp(a_i + b_i)$, i.e., $\mathsf{P}(l < \exp(a_i + b_i) < u) = p$ (Fong, Rue, and Wakefield 2010). For example, we can say we want $\mathsf{P}(0.1 < \exp(a_i + b_i) < 10) = 0.9$. This corresponds to a 90% credible interval $[0.1, 10]$ for $\exp(a_i + b_i)$. The idea is the same for a Poisson likelihood with a log link: we can think on the effect of the random effects on the relative risk.

Notation

We introduce new notation for the variance proportions and sums of variances. This notation is used throughout the remainder of the paper unless otherwise stated. Let $\boldsymbol{a}$ and $\boldsymbol{b}$ be two random effects in a latent Gaussian model, then we denote $\sigma^2_{\mathrm{a}} + \sigma^2_{\mathrm{b}} = \sigma^2_{\mathrm{a+b}}$ as the sum of the variances of the two random effects. The variance proportion describing how much of $\sigma^2_{\mathrm{a+b}}$ is attributed to $\boldsymbol{a}$ is denoted $\frac{\sigma^2_{\mathrm{a}}}{\sigma^2_{\mathrm{a}}+\sigma^2_{\mathrm{b}}} = \frac{\sigma^2_{\mathrm{a}}}{\sigma^2_{\mathrm{a+b}}} = \omega_{\frac{\mathrm{a}}{\mathrm{a+b}}}$. The amount of σ^2_{a+b} attributed to $\boldsymbol{b}$ is then $1 - \omega_{\frac{\mathrm{a}}{\mathrm{a+b}}} = \frac{\sigma^2_{\mathrm{b}}}{\sigma^2_{\mathrm{a+b}}}$.

In the genomic model in Example 2.1 with prior tree structure as in Figure 1a, the total variance (phenotypic variance) is given by $\sigma^2_{\mathrm{P}} = \sigma^2_{\mathrm{a}} + \sigma^2_{\varepsilon}$. With the notation we introduce now, this is denoted $\sigma^2_{\mathrm{a}+\varepsilon} = \sigma^2_{\mathrm{a}} + \sigma^2_{\varepsilon}$. The heritability h^2 is defined as the amount of phenotypic variance explained by the additive genetic effect, i.e., $h^2 = \frac{\sigma^2_{\mathrm{a}}}{\sigma^2_{\mathrm{a}}+\sigma^2_{\varepsilon}}$. With the new notation, this is $\omega_{\frac{\mathrm{a}}{\mathrm{a}+\varepsilon}}$.

3.3. Priors for variance proportions

Details about the prior distributions discussed in the previous section are presented here. While the **makemyprior** package can be used without knowing these details, they are needed to fully understand the hierarchical decomposition (HD) prior framework.

Shrinkage priors for variance proportions

We construct the joint hierarchical decomposition (HD) prior using a bottom-up approach following the prior tree, and the prior will thus be dependent on prior tree structure. The distance measure $d(\cdot)$ for a variance proportion depends on the covariance matrices of the effects of the child nodes in a split (Fuglstad *et al.* 2020), and the covariance matrix of a split node will be a function of the variance proportions(s) involved in this split. This means we must condition on the variance proportions associated with splits lower in the tree (if any), and that each prior depends on choices and covariance matrices at that and lower levels.

We omit the dependence of tree structure, covariance matrices and prior choices for other splits in the notation of the PC prior for readability. Consider a random intercept model $y_{i,j} = a_i + \varepsilon_{i,j}$ for $i, j = 1, \ldots, 10$, where $a_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_{\text{a}}^2)$ is a group effect and $\varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\varepsilon^2)$ is a residual effect. We define the variance proportion $\omega = \frac{\sigma_{\text{a}}^2}{\sigma_{\text{a}}^2 + \sigma_\varepsilon^2}$. Then we denote the different PC prior distributions as:

- $\sigma_* \sim \text{PC}_0(U, \alpha)$, with $\mathsf{P}(\sigma_* > U) = \alpha$, and shrinkage towards $\sigma_* = 0$.
- $\omega \sim \text{PC}_0(m)$ with $\mathsf{P}(\omega > m) = 0.5$ so that m defines the median, and shrinkage towards $\omega = 0$, i.e., the base model is a model with only $\boldsymbol{\varepsilon}$.
- $\omega \sim \text{PC}_1(m)$ with $\mathsf{P}(\omega > m) = 0.5$ so that m defines the median, and shrinkage towards $\omega = 1$, i.e., the base model is a model with only $\boldsymbol{a}$.
- $\omega \sim \text{PC}_{\text{M}}(m, c)$ with $\mathsf{P}(\omega > m) = 0.5$ and $\mathsf{P}(\text{logit}(1/4) < \text{logit}(\omega) - \text{logit}(m) < \text{logit}(3/4)) = c$ so that m defines the median, and c says something about how concentrated the distribution is around the median. The shrinkage is towards $\omega = m$, i.e., the base model is a combination of the effects $\boldsymbol{a}$ and $\boldsymbol{\varepsilon}$.

Note that $\text{PC}_1(m)$ for ω is equivalent to $\text{PC}_0(1 - m)$ for $1 - \omega = \frac{\sigma_\varepsilon^2}{\sigma_{\text{a}}^2 + \sigma_\varepsilon^2}$. Since the PC prior is a prior put on the distance between two models, and then transformed to the parameter of interest, we do not distinguish the notation between the PC prior on a standard deviation and variance parameter, as it will result in the same prior.

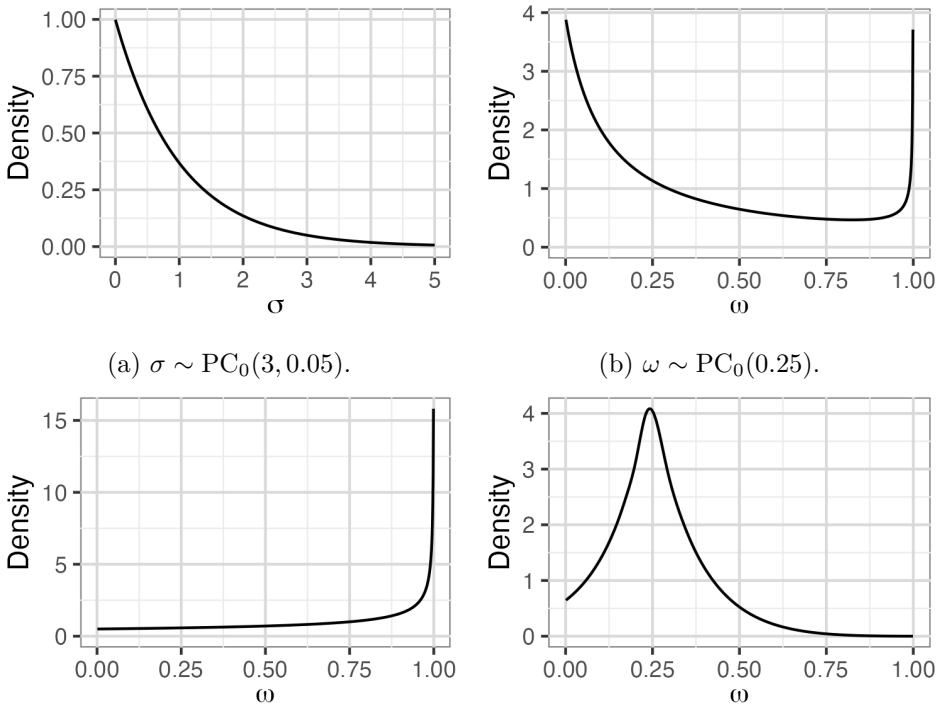
In Figure 2 we show examples for the four different priors. The shape of the PC prior on a standard deviation, shown in Figure 2a, is independent of the hyperparameters and other hyperparameters will simply give a rescaling of the axes.

Figure 2b shows a prior where we have shrinkage towards $\omega = 0$, which corresponds to a model with only residual variance. The spike at $\omega = 1$ is a consequence of the exponential prior on distance, and does not induce shrinkage to $\omega = 1$. See Guo *et al.* (2017) and Hem *et al.* (2021) for details.

In Figure 2c we show a prior where with shrinkage towards $\omega = 1$, giving a model with only group effect. This base model has a singular covariance matrix, giving a distance measure that is infinite. In such cases we cannot have a median that is further than 0.25 from the base model (see Fuglstad *et al.* 2020, Theorem 1 for details). In this case that means that the median cannot be smaller than 0.75.

The covariance matrix of the base model is always non-singular for a PC_{M} prior (shown in Figure 2d). The concentration parameter c in $\text{PC}_{\text{M}}(m, c)$ measures how certain we are about the prior median. A concentration of less than 0.5 will indicate that a prior with median at $\omega = 0.5$ is less concentrated around 0.5 than a uniform distribution on $[0, 1]$ would be, and we set the lower limit of c to 0.5. As this parameter says something about how much of the distribution mass is in an interval that is smaller than the parameter space ($0 \leq \omega \leq 1$), the distribution will not change much when c approaches 1.

The difference between the PC_0/PC_1 and PC_{M} priors is how certain the user is about the prior knowledge. In the genomic model in Example 2.1, a geneticist told us that the heritability h^2 is around 0.4. If the contribution of the additive effect is unclear, for example due to a small data sample, we can use $\text{PC}_0(0.4)$ for h^2, which has shrinkage towards 0 (only residual effect) and median 0.4. If we are certain that the additive genetic effect is contributing to



(a) $\sigma \sim \text{PC}_0(3, 0.05)$.

(b) $\omega \sim \text{PC}_0(0.25)$.

(c) $\omega \sim \text{PC}_1(0.75)$. Note that the base model has a singular covariance matrix.

(d) $\omega \sim \text{PC}_\text{M}(0.25, 0.85)$.

Figure 2: Examples of the different PC priors for a random intercept model $y_{i,j} = a_i + \varepsilon_{i,j}$ for $i, j = 1, \ldots, 10$. $\omega = \sigma_\text{a}^2/(\sigma_\text{a}^2 + \sigma_\varepsilon^2)$.

the variation in the observed phenotype, we use a PC_M prior with median $m = 0.4$, and we choose the concentration parameter c based on how strongly we believe in this value. This gives a prior with shrinkage towards a heritability of 0.4.

Ignorance priors for variance proportions

If we want to express ignorance about the variance distribution in a split, we use a symmetric Dirichlet prior distribution. For a split with $p \geq 2$ children, this prior is given by:

$$\pi(\boldsymbol{\omega}; p) = \text{Dirichlet}(p) = \frac{\Gamma(p\alpha)}{\Gamma(\alpha)^p} \left(\prod_{i=1}^{p} \omega_i \right)^{\alpha - 1},$$

where $\boldsymbol{\omega}$ is the vector of variance proportions involved in the split with $0 < \omega_i < 1$ for $i = 1, \ldots, p$ and $\sum_{i=1}^{p} \omega_i = 1$. Further, $\Gamma(\cdot)$ denotes the gamma function and $\alpha > 0$ is chosen so $\text{P}(\text{logit}(1/4) < \text{logit}(w_i) - \text{logit}(1/p) < \text{logit}(3/4)) = 1/2$ for $i = 1, \ldots, p$ (if this is achieved for one i, it is by symmetry achieved for all i). This prior assigns equal amount of variance to each model component. For a split consisting of p nodes we denote this as $\boldsymbol{\omega} \sim \text{Dirichlet}(p)$ for each proportion in the split. A multi-split is by itself a way of showing ignorance, and all multi-splits are given this Dirichlet prior.

Shrinkage priors for multi-splits

There may be situations where the user wants to assign unequal amounts of variance to the components in a multi-split. In that case, the user has opinions about the variance decomposition in the split, and an ignorant multi-split with a Dirichlet prior is not suitable. Instead we transform the multi-split to several dual splits and assign a PC prior to each of the dual splits. In Example 2.2, assume that we want a (20, 30, 50) division of the total variance of $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. We achieve this by first splitting the variance 50/50 between $\boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ with a $\text{PC}_\text{M}(0.5, c)$ prior, and then dividing the variance of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ 40/60 with a $\text{PC}_\text{M}(0.4, c)$ prior for some suitable value of c. The corresponding tree structure is shown in Figure 1c. We could also have chosen another effect to split off first, but Fuglstad *et al.* (2020) show that this order does not have much impact on the resulting prior.

Pre-computing the marginal priors

To ease the computation of the joint PC prior, we follow Fuglstad *et al.* (2020) and condition on the base models of the splits below in the tree, instead of on the parameters themselves. In this way we can pre-compute the marginal priors for each split in the tree. The base model for the Dirichlet distribution is equal variance to each component, i.e., for a split with p components the base model is $1/p$ of the variance in the split node to each child node. A split with a Dirichlet prior does not use information from lower levels in the tree.

4. Software overview

The R package **makemyprior** covers a wide range of Bayesian hierarchical models and will be of interest for a range of users. It currently supports latent Gaussian models where the observation model has a Gaussian likelihood with identity link, a Poisson likelihood with log link or a binomial likelihood with logit link. Supported latent Gaussian models can contain fixed and random effects. Random effects can be independent and identically distributed, i.e., $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$, follow a random walk of first or second order (corresponding to models `rw1` and `rw2` in **INLA**, Rue *et al.* 2009), see definitions in Rue and Held (2005, Sections 3.3.1 and 3.4.1, respectively), follow the Besag area model (model `besag` in **INLA**), see definition in Rue and Held (2005, Sections 3.3.2), or be user-defined latent Gaussian models controlled by one variance parameter σ^2 which follow the form $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{Q}^{-1})$, where $\mathbf{Q}$ is a user-defined symmetric positive definite structure matrix. For intrinsic Gaussian Markov random field models we apply the scaling proposed by Sørbye and Rue (2014).

As mentioned earlier fixed effects are of course supported in the linear predictor, but not integrated into the joint prior framework. By default a zero-mean normal prior with a fixed large variance is assigned. For variance parameters of a random effect or a total variance, PC priors, half-Cauchy priors, half-normal priors, inverse-gamma priors, or in the case of a total variance parameter also Jeffreys' prior are supported. Variance proportion parameters can follow the priors introduced in Section 3.3, namely PC_0, PC_1, PC_M or a symmetric Dirichlet distribution.

We mention two recent applications for which **makemyprior** might have come in handy. Slater, Brown, Rosenthal, and Mateu (2022) considered an augmented Besag-York-Mollié (BYM) model with a fixed effect, two Besag area models and an unstructured spatial component. The authors reparameterized the model analogously to Riebler, Sørbye, Simpson, and Rue (2016)

and placed a PC prior on the combined spatial variance and a symmetric Dirichlet prior on the weights which distribute the combined spatial variance to the random effect components. The package **makemyprior** is directly applicable to this model and allows for an equivalent prior derivation. Importantly, it makes it easily feasible and intuitive to incorporate even more prior knowledge on how the variation shall be distributed to the different spatial model components. Franco-Villoria, Ventrucci, and Rue (2020) apply the framework of Fuglstad *et al.* (2020) to the specific example of variance-partitioning in spatio-temporal disease mapping models. Here, the uncertainty is distributed between temporal effects, spatial effects and space-time interactions. The joint prior the authors propose corresponds to defining an underlying prior tree with two levels: At the upper level the total variance is first distributed between the interaction component, and the space-, time-main-effects, and then at the lower level the uncertainty assigned to the main effects is distributed between space and time. This model can be further extended by using structured and unstructured main effects for time and space. The package **makemyprior** can handle these models directly, and will formulate a joint prior based on the user-specific input. The prior will be computed without further involvement of the user and can be visualized for inspection.

Posterior inference can be directly computed through **INLA** and **rstan**. Additionally, we provide functions to evaluate the joint prior generated by **makemyprior** at arbitrary parameter values. This is suitable for inclusion of the priors in other R packages or self-written code for inference.

5. The makemyprior package

Throughout this section, the use of **makemyprior** is exemplified by the following model.

Model 1 (Example model)**.** Consider the hierarchical model for the $n = m \cdot p$ observations $y_{i,j}$, $i = 1, \dots p$ and $j = 1, \dots, m$, given by

$$\begin{aligned} y_{i,j} | \eta_{i,j}, \sigma_\varepsilon^2 &\sim \mathcal{N}(\eta_{i,j}, \sigma_\varepsilon^2), \\ \eta_{i,j} &= \mu + x_i \beta + a_i + b_j, \end{aligned}$$

where μ is an intercept, x_i is a covariate with coefficient β, and $a_1, a_2, \dots, a_p \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\text{a}^2)$ and $b_1, b_2, \dots, b_m \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\text{b}^2)$ are random effects. $\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\varepsilon^2)$ are residuals.

The data for this model could be some measures made for year i and age group j with a time-specific covariate x_i. We do not interpret the meaning of the data in this section.

We include four different tree structures for this model in Table 1. Several likelihoods, latent models and prior distributions are available in **makemyprior**, which can be listed with the function:

```
makemyprior_models(type = c("prior", "latent", "likelihood"), select = NULL)
```

5.1. Specifying the linear predictor and preparing the data object

First, the linear predictor is specified using a formula object with a syntax similar to e.g., `lm()` from **stats** and `inla()` from **INLA**. Covariates are included directly by name in the formula, and `mc()` is used to include information about each random effect:

```
mc(label, model = "iid", ...)
```

The main arguments are `label` (name of the effect) and `model` (the type of latent model effect, i.i.d. is the default). The other arguments depend on the choice of latent model, see documentation for details.

For Model 1 where both $\boldsymbol{a}$ and $\boldsymbol{b}$ are i.i.d., the formula is:

```
R> formula <- y ~ x + mc(a) + mc(b)
```

The intercept μ is included by default, but can be removed with `-1`. The residual effect for a Gaussian likelihood should not be specified in the formula.

The second step is to gather data and create a data object as either a `data.frame` or `list` with names corresponding to the elements of the formula. For Model 1, we need the observed response `y`, the covariate `x`, and indexes for `a` and `b` (these must be specified as integers). A simple simulated dataset is:

```
R> p <- 10
R> m <- 10
R> n <- m * p
R> set.seed(1)
R> data <- list(a = rep(1:p, each = m),
+    b = rep(1:m, times = p), x = runif(n))
R> data$y <- data$x + rnorm(p, 0, 0.5)[data$a] +
+    rnorm(m, 0, 0.3)[data$b] + rnorm(n, 0, 1)
```

We recommend the use of short names for the input data because these names need to be used to refer to model components in later steps. An example is `"rain"` instead of `"rainfall_august_2020"`. Note that the observations $y_{i,j}$ are not used to make the prior.

5.2. Exploring and selecting the prior graphically

We provide a graphical user interface (GUI) where the user can construct the prior in an interactive way. The GUI is implemented as a **shiny** app (Chang *et al.* 2024) running locally. It allows the user to first define the tree structure, and then be guided sequentially through the steps of selecting priors for each split, singleton and top node. The **shiny** apps are useful to, for example, display and investigate results from analyses. For example, the user can customize graphs and tables in a simple way. Such apps are used by Depaoli, Winter, and Visser (2020) to show why prior sensitivity analysis is important, and by Smid and Winter (2020) to let users explore the impact of prior distributions in inference. However, to the extent of our knowledge, as of today there are no packages or apps that allows the user to use a **shiny** app (or similar) to specify priors for custom models and data and directly carry out inference.

The first step is to initialize a prior object. This is done with the function `make_prior()`:

```
make_prior(formula, data, family = "gaussian", prior = list(),
  intercept_prior = c(), covariate_prior = list())
```

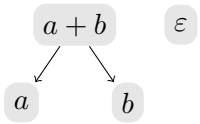
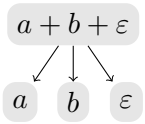
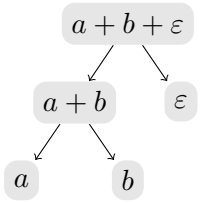
	Tree structure	Text string
Prior 1		`"(a); (b); (eps)"`
Prior 2		`"s1 = (a, b); (eps)"`
Prior 3		`"s1 = (a, b, eps)"`
Prior 4		`"s1 = (a, b); s2 = (s1, eps)"`

Table 1: Four tree structures for Model 1 with text strings specifying them. The names `s1` and `s2` on the splits (variance proportions) are chosen by the user in the initial specification of the prior, and are used in a nested formulation to specify the prior tree structure.

`formula` and `data` are the objects created in Section 5.1, and `family` is the likelihood (`"gaussian"` is the default, `"binomial"` and `"poisson"` are also available). The `prior` argument is not relevant when the GUI is used, and its description is deferred to Section 5.3. `intercept_prior` and `covariate_prior` specifies the mean and standard deviation for the Gaussian priors on the intercept and covariate coefficients. The default in **makemyprior** is $\mathcal{N}(0, \sigma = 1000)$ for both, and the coefficient priors are specified as a named list with names corresponding to the covariate names.

For Model 1, we can create a prior object with the following command:

```
R> prior <- make_prior(
+    formula, data, family = "gaussian",
+    intercept_prior = c(0, 1000),
+    covariate_prior = list(x = c(0, 100)))

Warning message:
Did not find a tree, using default tree structure instead.
```

This gives a prior with a single tree with one split as shown in Prior 3 in Table 1, which is the default setting. A warning makes the user aware that the default prior is chosen. For Gaussian likelihoods, Jeffreys' prior is by default set for the total variance. Let $\sigma^2_{\text{a+b+}\varepsilon}$ denote total variance, and $\boldsymbol{\omega} = (\omega_{\frac{\text{a}}{\text{a+b+}\varepsilon}}, \omega_{\frac{\text{b}}{\text{a+b+}\varepsilon}}, 1 - \omega_{\frac{\text{a}}{\text{a+b+}\varepsilon}} - \omega_{\frac{\text{b}}{\text{a+b+}\varepsilon}})$ describe the attribution of variance to the three different sources, then the initial choice of priors is:

$$\boldsymbol{\omega} \sim \text{Dirichlet}(3) \quad \text{and} \quad \sigma^2_{\text{a+b+}\varepsilon} \sim \text{Jeffreys}'. \tag{1}$$

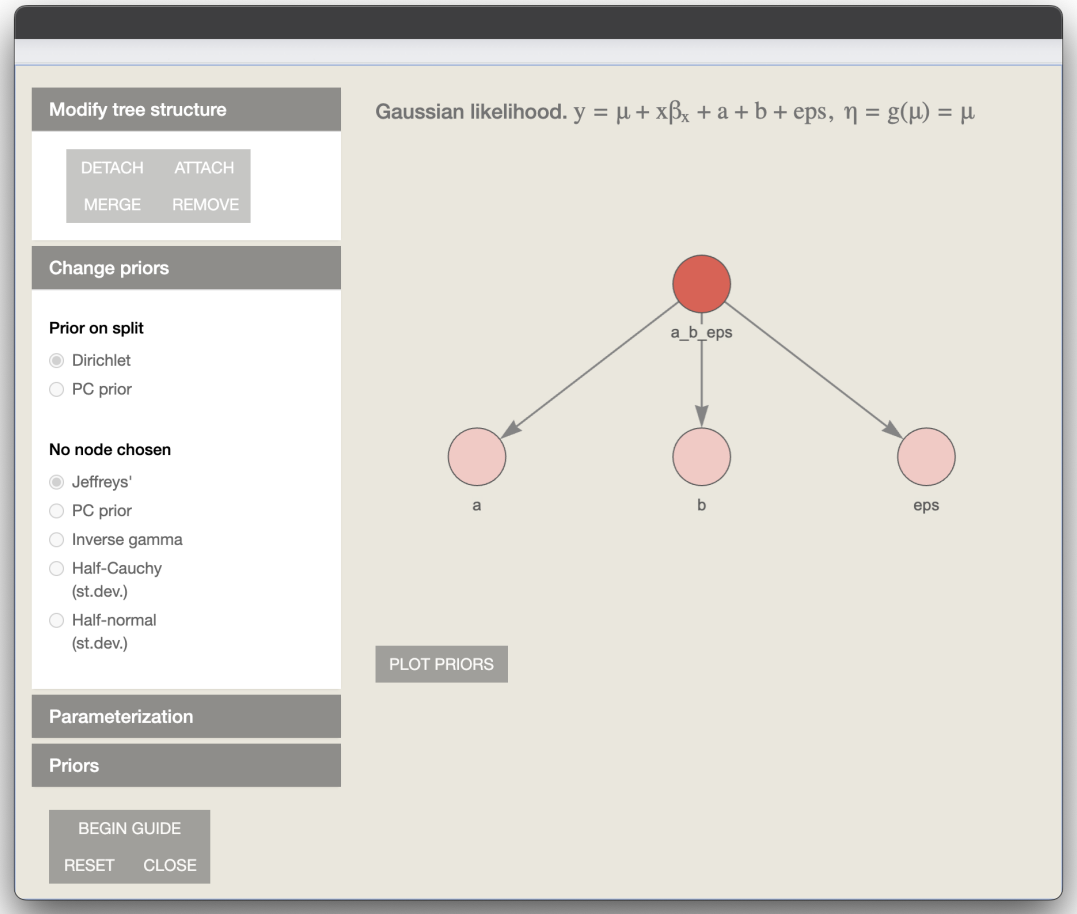


Figure 3: Screenshot of the GUI in **makemyprior** with the default prior (Equation 1).

The intercept has a $\mathcal{N}(0, \sigma = 1000)$ prior, and the covariate β a $\mathcal{N}(0, \sigma = 100)$ prior.

The function `makemyprior_gui()` allows the user to select the desired prior tree structure and choose prior distributions interactively:

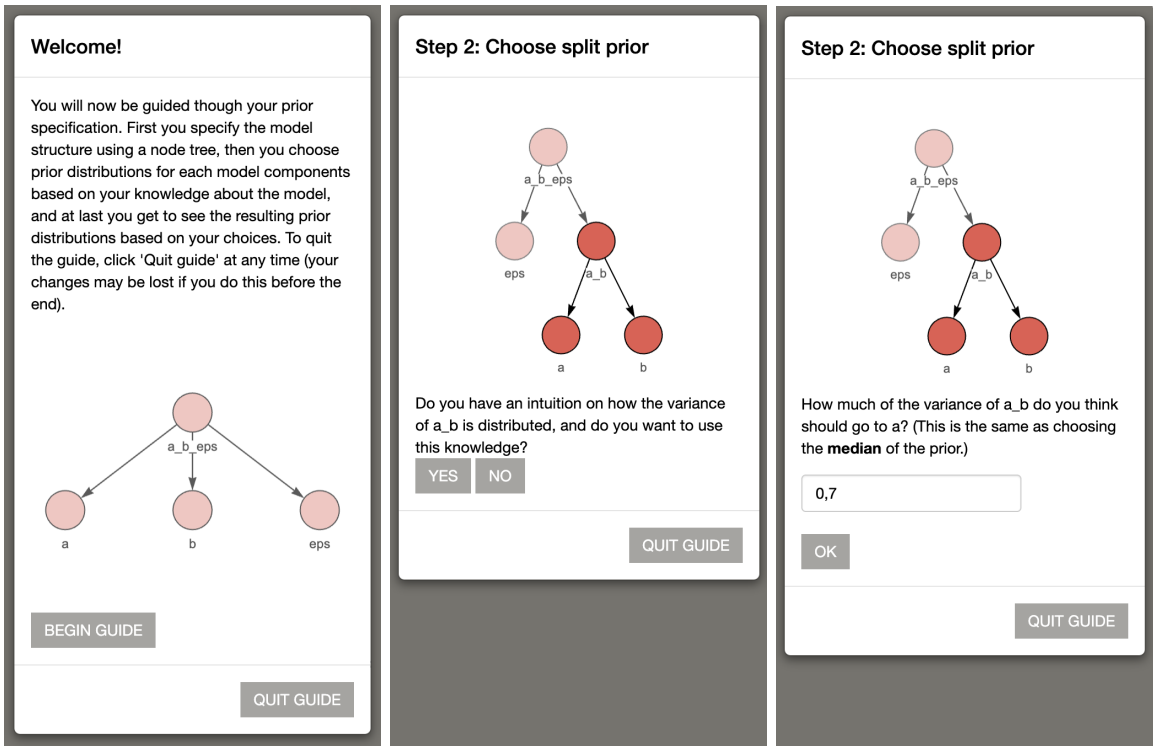
```
makemyprior_gui(prior, guide = FALSE, no_pc = FALSE)
```

This function takes the arguments `prior`, created with `make_prior()` earlier, `guide` which specifies whether or not the guide should automatically start (the guide can be started at any time), and `no_pc`. The PC prior is computed using the covariance matrix structure of the model components, which may be slow for large models. For a better user experience, the user can turn off the computation of the PC prior in the GUI using `no_pc = TRUE`. The prior will be computed upon closing, and this will only affect the plotting of the prior in the GUI.

For Model 1, we start the GUI by running:

```
R> new_prior <- makemyprior_gui(prior)
```

This saves the new prior chosen to the variable `new_prior` when we close the GUI. Figure 3 shows a screenshot of the GUI for Model 1 for $m = p = 10$ and the default prior. By selecting



(a) Initial window. (b) Question for prior elicitation. (c) Question for prior elicitation.

Figure 4: Screenshots from the guide in the **makemyprior** GUI.

nodes and using the buttons located in the panels to the left, the user can create the desired tree structure and choose the desired priors for this tree.

The user can click "Begin guide" to the lower left to be guided through the prior construction. Figure 4a shows the initial window of the guide, which consists of two main steps: First, the user is helped to create the desired tree structure. Note that every time the tree structure is modified, both when using the guide and otherwise, all splits are set to have the default Dirichlet prior. After deciding on a tree structure, the user will be taken through the prior tree structure in a step-wise manner. For each split, top node and singleton they will be asked simple questions about the existing prior knowledge about the corresponding parameter. This eases the process of using the knowledge to create the prior, as the user does not need to think in terms of prior distributions directly, but only needs to answer questions such as shown in Figure 4b and 4c. After modifying the prior to have the following distributions:

$$\omega_{\frac{a}{a+b}} \sim \mathrm{PC_M}(0.7, 0.5),\ \omega_{\frac{a+b}{a+b+\varepsilon}} \sim \mathrm{PC_0}(0.25), \text{ and } \sigma_{a+b+\varepsilon} \sim \mathrm{PC_0}(3, 0.05), \tag{2}$$

the GUI looks like the screenshot in Figure 5. The chosen prior distributions and the connection between the parameterization and model variances are easily seen to the left.

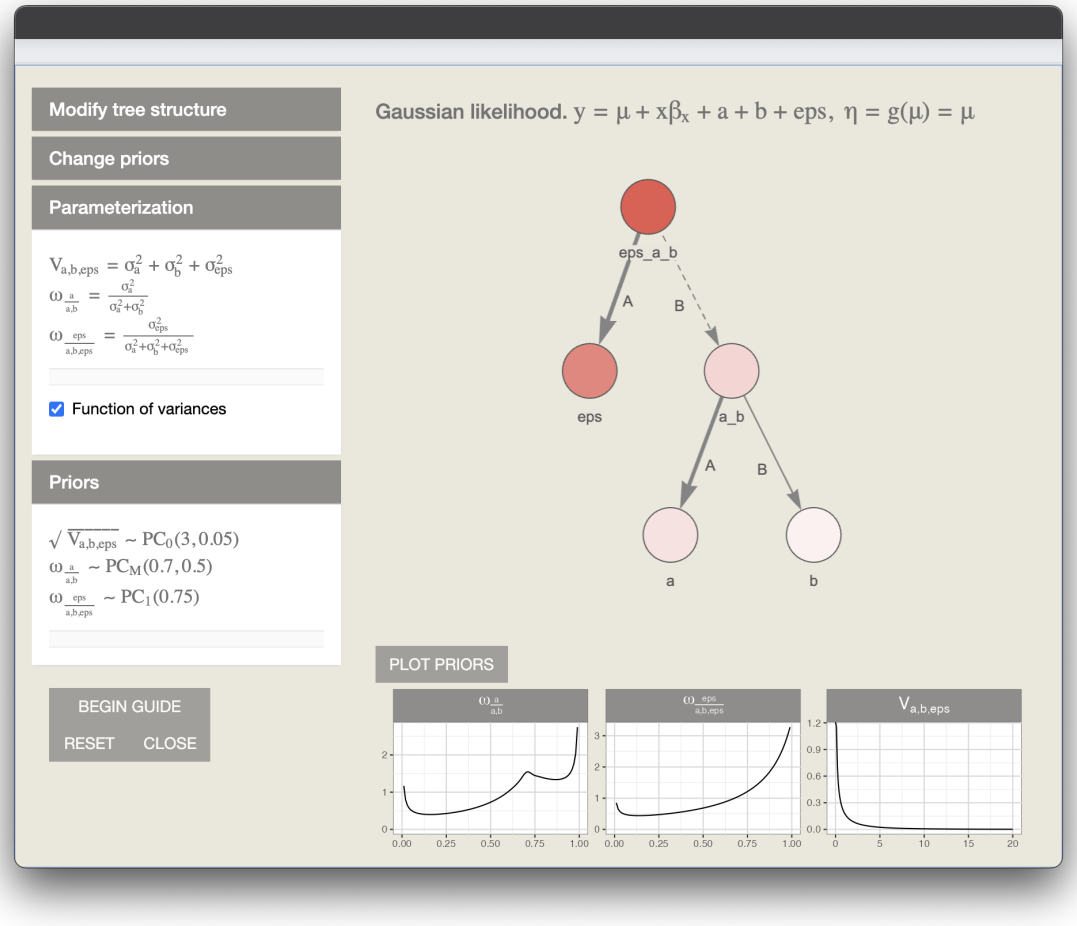


Figure 5: Screenshot of the GUI in **makemyprior** when the prior in Equation 2 is chosen.

The GUI is intuitive and contains a thorough description of the options, and we do not explain the features in detail here. We instead recommend using the guide in the GUI to get familiar with the package. A summary of the prior object can be printed with:

```
R> summary(new_prior)
```

```
Model: y ~ x + mc(a) + mc(b)
Tree structure: a_b = (a,b); eps_a_b = (eps,a_b)

Weight priors:
        w[a/a_b] ~ PCM(0.7, 0.5)
        w[eps/eps_a_b] ~ PC1(0.75)
Total variance priors:
        sqrt(V)[eps_a_b] ~ PC0(3, 0.05)

Covariate priors: intercept ~ N(0, 1000^2), x ~ N(0, 100^2)
```

5.3. Selecting the prior non-graphically

If the prior has been constructed with the GUI, this section can be skipped. Here we describe an alternative way to specify the prior without the use of the GUI. This is done with same function we use to make the default prior, `make_prior()`, by specifying the `prior` argument. Argument `prior` is a named `list` with the following arguments:

`tree` The tree structure as a string. A split is specified as `s1 = (a, b)`, where `s1` represents a split node and can be any name except names of the input data in `data` and the reserved `eps`, which is used for residuals for a Gaussian likelihood. Short names are recommended. Note that these split names are just used in the initial specification. The child nodes for each split are included in parentheses separated by commas, and each split is separated by semicolons. Singletons are included as `(a)`. Examples of strings for different tree structures for Model 1 are shown in Table 1.

`V` A named list with information on the priors on each top node and singleton, i.e., all variances. Options are `"pc"`, `"jeffreys"`, `"invgam"`, `"hn"` (half-normal) and `"hc"` (Half-Cauchy). The names in the list are the top node and singleton names from the `tree` argument.

`w` A named list with information on the priors on each split, i.e., all variance proportions. The names in the list are the split node names from the `tree` argument. Options are `"pc0"`, `"pc1"`, `"pcM"` and `"dirichlet"`.

`V` and `w` must have the following structure for each element in the list: `list(prior = prior_name, param = parameter_vector)`, except for Jeffreys' and Dirichlet priors, where `param` is not specified (as the distributions do not have hyperparameters). See Section 6 for how to specify priors in specific examples, and `makemyprior_models()` for more details about the different prior distributions.

The prior in Equation 2 for Model 1 can be specified as:

```
R> prior <- make_prior(
+    formula, data,
+    prior = list(
+      tree = "s1 = (a, b); s2 = (s1, eps)",
+      w = list(s1 = list(prior = "pcM", param = c(0.7, 0.5)),
+               s2 = list(prior = "pc0", param = 0.25)),
+      V = list(s2 = list(prior = "pc0", param = c(3, 0.05)))
+    ),
+    covariate_prior = list(x = c(0, 100)))
R> prior

Model: y ~ x + mc(a) + mc(b)
Tree structure: a_b = (a,b); eps_a_b = (eps,a_b)

Weight priors:
        w[a/a_b] ~ PCM(0.7, 0.5)
        w[eps/eps_a_b] ~ PC1(0.75)
```

```
Total variance priors:
        sqrt(V)[eps_a_b] ~ PC0(3, 0.05)
```

The names `s1` and `s2` are chosen by the user in the initial specification of the prior, and can be any names that are not used for the data or `eps` (reserved for residuals). Note that `s1` and `s2` are only used as a link between the splits and priors, and have been automatically changed to `a_b` and `eps_a_b` by `make_prior()`. The order we list the children for each split node in the `tree` argument decides which way we have shrinkage with the PC priors. For `s1 = (a, b)`, $\text{PC}_0(m)$ shrinks effect $\boldsymbol{a}$ ($\omega_{\frac{a}{a+b}} = 0$ as base model), $\text{PC}_1(m)$ shrinks effect $\boldsymbol{b}$ and $\text{PC}_\text{M}(m, c)$ gives shrinkage towards $m\boldsymbol{a} + (1-m)\boldsymbol{b}$. All three has median at $\omega_{\frac{a}{a+b}} = m$. Note that $\text{PC}_0(m)$ on $\omega_{\frac{a}{a+b}}$ is equivalent to $\text{PC}_1(1-m)$ on $\omega_{\frac{b}{a+b}}$.

All top nodes, singletons and split nodes without a specified prior will get the default prior. The default settings in **makemyprior** are chosen based on the findings of Fuglstad *et al.* (2020) to ensure robust inference:

- If no prior is specified (neither tree structure nor priors), the prior will be a joint prior where all latent components (including a possible residual effect) get an equal amount of the total variance in the prior through the symmetric Dirichlet prior, and the default total variance prior.
- The default prior on the total variance (top node) varies with likelihood:
  - Jeffreys' prior for Gaussian likelihood for a tree structure with one tree, $\text{PC}_0(3, 0.05)$ otherwise.
  - $\text{PC}_0(1.6, 0.05)$ for binomial likelihood.
  - $\text{PC}_0(1.6, 0.05)$ for Poisson likelihood.
- The default prior on an individual variance (singleton) varies with likelihood:
  - $\text{PC}_0(3, 0.05)$ for Gaussian likelihood.
  - $\text{PC}_0(1.6, 0.05)$ for binomial likelihood.
  - $\text{PC}_0(1.6, 0.05)$ for Poisson likelihood.
- The default prior on a variance proportion (split node) is a Dirichlet prior assigning equal amount of variance to each of the model components involved in the split.

The reasoning behind these choices are as follows. For the variance proportions, there is no way of knowing what behavior is desired in general. Therefore we use the ignorant Dirichlet prior as the default. A standard Gaussian distribution with mean 0 and variance 1 will have close to all the density mass between -3 and 3, and $\text{PC}_0(3, 0.05)$ is a vague prior for the standard deviation in such a distribution. We use Jeffreys' prior when applicable (see Section 3.2). To choose the default variance (standard deviation) prior for other likelihoods, we follow the idea of Fong *et al.* (2010) and use a credible interval on a suitable scale. The default prior for all variance parameters (both for top nodes and singletons) in **makemyprior** for binomial and Poisson likelihoods is a PC prior with a 95% credible interval between 0.2 and 5 for the multiplicative effect on the odds ratio and risk, respectively. This is obtained with a $\text{PC}_0(1.6, 0.05)$ prior. We want to emphasize that before selecting the default prior, both

when using **makemyprior** and otherwise, you should take a moment and consider whether or not it is suitable for your model and data.

5.4. Performing inference

We include functions for inference that are compatible with the prior object obtained from `make_prior()` (and `makemyprior_gui()`). Both Stan (Carpenter *et al.* 2017) through **rstan** (Stan Development Team 2024) and integrated nested Laplace approximations (INLA, Rue *et al.* 2009) through **INLA** (see `https://www.R-INLA.org/`) can be used for the inference.

Stan is a probabilistic programming language, where Hamiltonian Monte Carlo (HMC) is used to sample from the posterior distribution (Carpenter *et al.* 2017). Stan implements HMC using the No U-Turn Sampler (NUTS, Hoffman and Gelman 2014). NUTS reduces the need for tuning of the sampler, making it easy to use as no manual settings are needed for the algorithm to run, and the user only needs to provide the joint prior and likelihood model, implemented in a programming language similar to C++. We provide Stan code for fitting latent Gaussian models with certain likelihoods and latent effects. The internal parameterization in the provided Stan code is log-variance, and the prior is transformed from the parameterization given by the prior tree structure to log-variances.

INLA is a non-sampling based method for doing fast and efficient Bayesian inference on latent Gaussian models (Rue *et al.* 2009), utilizing Gaussian Markov random fields (GMRFs) with sparse precision matrices, which gives computational benefits through the Markov property. The INLA method approximates the posteriors by a mixture of Gaussian distributions and applies a skewness correction to the marginals (Rue, Riebler, Sørbye, Illian, Simpson, and Lindgren 2017). It is easy and straight-forward to use for inference, and can fit models with a broad range of latent effects. The internal parameterization of the model parameters in **INLA** is log-precision, and in the same way as for the provided Stan code the parameterization following the prior tree is transformed to fit **INLA**.

Some common latent models are included in the code for the package: i.i.d. (`"iid"`), random walk of first (`"rw1"`) and second (`"rw2"`) order, Besag (`"besag"`), and effects with structured covariance matrices (`"generic0"`). The likelihood family was specified in `make_prior()`. In Section 6 we show how the inference can be performed. Here we describe the functions that can be used for inference.

Inference with Stan

Stan is a flexible tool for inference, however, it requires the user to write custom code for the model that is to be fitted. **makemyprior** contains pre-written Stan code that can be used to do inference on latent Gaussian models with a selection of latent models. We recommend to compile the Stan code before doing inference with Stan. This can be done with the following function:

```
compile_stan(save = FALSE, permanent = FALSE, path = NULL)
```

where `save` indicates whether or not to save the compiled object (must be set this to `TRUE` to avoid recompiling the code every time inference is performed), and `permanent` is set to `FALSE` if the compiled model should be saved temporary in `tempdir()` for the current R session, or permanently in the package directory. `path` is only necessary if the default location for saving

the compiled object is not possible to use (see `?compile_stan` for details). For inference with Stan we use the following function:

```
inference_stan(prior_obj, use_likelihood = TRUE, print_prior = TRUE,
               path = NULL, ...)
```

The first argument is the prior object from `make_prior()` or `makemyprior_gui()`. The user can specify whether to include the likelihood (`use_likelihood = TRUE`) or not (`FALSE`). In the latter case we sample from the prior distribution. `print_prior` (`TRUE` by default) prints details about the chosen prior. `path` is a file path that can be specified if the `path` argument were used in `compile_stan()`, and if left empty, `inference_stan()` looks for the compiled Stan code in `tempdir()`, then the package directory, and if no compiled code is found the function re-compiles the Stan code. Additional arguments that are sent directly to the **rstan** function `sampling()` can be specified for the inference. Useful arguments include `iter` (total number of iterations for each chain), `warmup` (number of iterations for the warm-up), `chains` (number of chains), `seed` (for reproducibility), and `control` (for specifying algorithm tuning parameters).

The internal parameterization in the Stan code included in the package is log-variance, however, since Stan works with samples we can look at any parameterization we want by transforming the log-variances. For using other latent models or more complex models than the ones provided in the included Stan code (see above), the user must write customized Stan code, see Section 5.6.

*Inference with **INLA***

For inference with **INLA** we use the following function:

```
inference_inla(prior_obj, use_likelihood = TRUE, print_prior = TRUE, ...)
```

The first three arguments are the same as in `inference_stan()`. Additional arguments can be fed to the **INLA** function `inla()`. Useful arguments include `Ntrials` for the binomial likelihood, used to specify the amount of trials, where the response is the number of successes.

5.5. Visualizing priors and posteriors

We offer several functions to visualize the prior and posterior distributions. The prior distributions for the random effects on the tree structure parameterization can be plotted with `plot_prior(obj)` which take an object from `make_prior()`, `makemyprior_gui()`, `inference_stan()` or `inference_inla()` as input. The posterior distributions can be displayed with

```
plot_posterior_variance(obj)
plot_posterior_stdev(obj)
plot_posterior_precision(obj)
```

`obj` is an object from `inference_stan()` or `inference_inla()`.

The posterior distributions of random effects from inference with Stan can be plotted with:

```
plot_posterior_stan(
  obj,
  param = c("prior", "variance", "stdev", "precision"),
  prior = FALSE
)
```

Here, `obj` is an object from `inference_stan()`, `param` specifies which parameterization the plots should have where `param = "prior"` gives the posterior on the same parameterization as the prior. `prior` indicates whether or not to plot the prior together with the posterior for `param = "prior"`. The total variance prior will only be plotted if it is not Jeffreys' prior. Fixed effect posteriors can be plotted with `plot_posterior_fixed(obj)`. More details about visualization can be found with `?makemyprior_plotting`.

5.6. More complex models in Stan

Latent models may have parameters that are not variances, such as correlations. These non-variance parameters are handled independently in the HD prior (Fuglstad *et al.* 2020). The Stan code included in **makemyprior** is applicable for certain commonly used latent models and likelihoods (see Section 5.4). We provide a "skeleton" code and a description on how the user can write custom Stan code and include the joint prior created with `make_prior()`. This can be accessed with:

```
create_stan_file(location = "")
```

`location` is a string to a path where a folder with necessary files will be stored. The user can edit the code and include custom latent components etc. We do not include details on this, as it will be highly model specific and is merely an offer to the users who want to apply the HD prior in more advanced models.

5.7. Evaluating the joint prior

The following functions allow users to construct the joint prior through **makemyprior**, and evaluate the priors in their own inference code. With the function `eval_joint_prior()`, the joint HD prior created with `make_prior()` can be evaluated on log-variance scale:

```
eval_joint_prior(theta, prior_data)
```

where `theta` is a vector of log-variances and `prior_data` is a condensed prior object created with `make_eval_prior_data`. The latter can also be used to extract the necessary data from a prior object such that it can be used with the `inla()` function from **INLA** directly, which can be useful when expert options with INLA is desired. The marginal prior distributions for PC priors on variance proportions can be evaluated using:

```
eval_pc_prior(x, obj, param, logitscale = FALSE)
```

where `x` is value(s) to evaluate the prior in, `obj` is an object from `make_prior()`, `param` is a string indicating which variance proportion we want to evaluate, and `logitscale` indicates whether the input `x` is on logit-scale (`TRUE`) or not.

6. Using makemyprior: Examples

In this section we provide three examples where we use the **makemyprior** package to construct priors and run inference. Two examples are with Gaussian responses, and one is with Binomial responses. We have used `Stan` for the inference (with `inference_stan()`), but the procedure is the same for inference with **INLA** (using `inference_inla()` instead).

6.1. Gaussian responses

Genomic selection in wheat breeding

This is an extended version of the model in Example 2.1. In addition to the additive genetic effect $\boldsymbol{a}$, we now also include two nonadditive effects: Dominance $\boldsymbol{d}$ and additive-by-additive epistasis $\boldsymbol{x}$. This example is taken from Hem *et al.* (2021). The response y_i is grain yield for individual i. We utilize the expert knowledge elicited from experts in the field, and create a prior distribution reflecting this knowledge. We model the response as:

$$y_i = \mu + a_i + d_i + x_i + \varepsilon_i, \ i = 1, \ldots, 100, \tag{3}$$

where μ is an intercept with default $\mathcal{N}(0, 1000^2)$ prior and ε_i is the residual effect, representing environmental noise. Further, a_i, d_i and x_i are additive, dominance and epistasis (additive-by-additive epistasis) effects, respectively. These three add up to the genetic effect $g_i = a_i + d_i + x_i$. We assume that $\boldsymbol{a} = (a_1, \ldots, a_{100}) \sim \mathcal{N}_{100}(\boldsymbol{0}, \sigma_{\text{a}}^2 \boldsymbol{A})$, $\boldsymbol{d} = (d_1, \ldots, d_{100}) \sim \mathcal{N}_{100}(\boldsymbol{0}, \sigma_{\text{a}}^2 \boldsymbol{D})$, and $\boldsymbol{x} = (x_1, \ldots, x_{100}) \sim \mathcal{N}_{100}(\boldsymbol{0}, \sigma_{\text{a}}^2 \boldsymbol{X})$, and we use a sum-to-zero constraint on all genetic effects. The covariance matrices $\boldsymbol{A}$, $\boldsymbol{D}$ and $\boldsymbol{X}$ are computed from the single nucleotide polymorphism (SNP) matrix with thousands of genetic markers, see Hem *et al.* (2021) for details. This model has structured covariance matrices, and we use the `"generic0"` latent model. `"generic0"` requires the argument `Cmatrix`, which is the precision (inverse covariance) matrix $\boldsymbol{Q}_*$ for the effect. With these data, we get the following formula:

```
R> formula <- y ~
+    mc(a, model = "generic0", Cmatrix = Q_a, constr = TRUE) +
+    mc(d, model = "generic0", Cmatrix = Q_d, constr = TRUE) +
+    mc(x, model = "generic0", Cmatrix = Q_x, constr = TRUE)
```

We go through the reasoning behind a prior where we use all available prior knowledge, following the tree structure in Table 2.

The expert in genetics has information on the heritability, which is the amount of total variance attributed to the genetic effects and on the distribution of the genetic effect $\boldsymbol{g}$ to the additive, dominance and epistasis effects $\boldsymbol{a}$, $\boldsymbol{d}$ and $\boldsymbol{x}$. The expert says the heritability $\omega_{\frac{\text{g}}{\text{g}+\varepsilon}}$ is around 0.25, and that we want to avoid overfitting, which leads us to $\omega_{\frac{\text{g}}{\text{g}+\varepsilon}} \sim \text{PC}_0(0.25)$. The additive, dominance and epistasis effects have according to the expert a division of the genetic variance that is around (85, 10, 5)%, respectively. To achieve this, we must use two dual-splits to decompose the genetic variation, and do this by splitting off the additive effect first, with a $\text{PC}_{\text{M}}(0.85, 0.8)$ prior on $\omega_{\frac{\text{a}}{\text{g}}}$ (the amount of genetic variance that is additive). Then we attribute the remaining 15% of the genetic variance to $\boldsymbol{d}$ and $\boldsymbol{x}$ with 67% to $\boldsymbol{d}$ with $\text{PC}_{\text{M}}(0.67, 0.8)$ on $\omega_{\frac{\text{d}}{\text{d}+\text{x}}}$. We choose a concentration parameter value of 0.8 because the expert

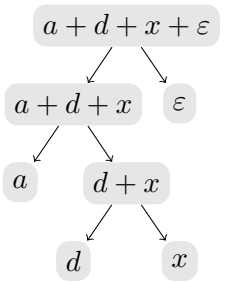
Tree structure	Parameters, priors
	$\sigma^2_{\text{a+d+x+}\varepsilon} \sim \text{Jeffreys}'$ $\omega_{\frac{\text{g}}{\text{g+}\varepsilon}} \sim \text{PC}_0(0.25)$ $\omega_{\frac{\text{a}}{\text{g}}} \sim \text{PC}_\text{M}(0.85, 0.8)$ $\omega_{\frac{\text{d}}{\text{d+x}}} \sim \text{PC}_\text{M}(0.67, 0.8)$

Table 2: Tree structures and the corresponding parameters for the genomic example: $g_i = a_i + d_i + x_i$.

is quite sure about the (85, 10, 5)% division. This corresponds to having 75% of the density mass in the interval $[\text{logit}(m) - 1, \text{logit}(m) + 1]$. The expert does not want to use expert knowledge for the total variance $\sigma^2_{\text{a+d+x+}\varepsilon}$, so we use Jeffreys' prior.

We have simulated a dataset following the description in Hem *et al.* (2021) (see also Gaynor *et al.* 2017; Selle *et al.* 2019), using the R package **AlphaSimR** (Faux *et al.* 2016; Gaynor 2023). The source code for simulating the dataset is available in the Supplemental Materials of Hem *et al.* (2021) (Hem, Selle, Gorjanc, Fuglstad, and Riebler 2020). The dataset is included as `wheat_data` in **makemyprior**. To incorporate the expert knowledge in a unified way, we first scale the covariance matrices to have typical variance equal to 1 (see Sørbye and Rue 2014, for details), using the function `scale_precmat()` in **makemyprior**:

```
R> wheat_data_scaled <- wheat_data
R> wheat_data_scaled$Q_a <- scale_precmat(wheat_data$Q_a)
R> wheat_data_scaled$Q_d <- scale_precmat(wheat_data$Q_d)
R> wheat_data_scaled$Q_x <- scale_precmat(wheat_data$Q_x)
```

This model is implemented as follows:

```
R> prior <- make_prior(formula, wheat_data_scaled, prior = list(
+    tree = "s1 = (d, x); s2 = (a, s1); s3 = (s2, eps)",
+    w = list(
+      s1 = list(prior = "pcM", param = c(0.67, 0.8)),
+      s2 = list(prior = "pcM", param = c(0.85, 0.8)),
+      s3 = list(prior = "pc0", param = 0.25))))
```

Note that we omit the specification of the total variance prior, as we choose the default Jeffreys' prior. It can be specified with adding `V = list(s3 = list(prior = "jeffreys"))` to the list provided to the `prior` argument. We now do inference on this model and plot the results:

```
R> posterior <- inference_stan(prior, iter = 15000, warmup = 5000,
+    seed = 1, init = "0", chains = 1)
R> plot_posterior_stan(posterior, param = "prior", prior = TRUE)
```

```
Tree structure: d_x = (d,x); a_d_x = (a,d_x); eps_a_d_x = (eps,a_d_x)

Weight priors:
        w[d/d_x] ~ PCM(0.67, 0.8)
        w[a/a_d_x] ~ PCM(0.85, 0.8)
        w[eps/eps_a_d_x] ~ PC1(0.75)
Total variance priors:
        V[eps_a_d_x] ~ Jeffreys'

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000184 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition
would take 1.84 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:     1 / 15000 [  0%]  (Warmup)
Chain 1: Iteration:  1500 / 15000 [ 10%]  (Warmup)
Chain 1: Iteration:  3000 / 15000 [ 20%]  (Warmup)
Chain 1: Iteration:  4500 / 15000 [ 30%]  (Warmup)
Chain 1: Iteration:  5001 / 15000 [ 33%]  (Sampling)
Chain 1: Iteration:  6500 / 15000 [ 43%]  (Sampling)
Chain 1: Iteration:  8000 / 15000 [ 53%]  (Sampling)
Chain 1: Iteration:  9500 / 15000 [ 63%]  (Sampling)
Chain 1: Iteration: 11000 / 15000 [ 73%]  (Sampling)
Chain 1: Iteration: 12500 / 15000 [ 83%]  (Sampling)
Chain 1: Iteration: 14000 / 15000 [ 93%]  (Sampling)
Chain 1: Iteration: 15000 / 15000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 9.302 seconds (Warm-up)
Chain 1:                19.386 seconds (Sampling)
Chain 1:                28.688 seconds (Total)
Chain 1:
```

Figure 6 shows the prior and posterior together on the parameterization of the prior (see Table 2).

We see that we do not have enough data to estimate the variance proportion for the additive and nonadditive genetic effects: The posterior distribution is almost identical to the prior distribution. Hem *et al.* (2021) have conducted an extensive simulation study on this and similar models. They saw a strong need for robust prior distributions, which we also see in Figure 6, because the nonadditive effects $\boldsymbol{d}$ and $\boldsymbol{x}$ are strongly confounded with the environmental effect $\boldsymbol{\varepsilon}$, and the number of observations is small compared to the number of genetic markers that needs to be estimated (Sorensen and Gianola 2007).

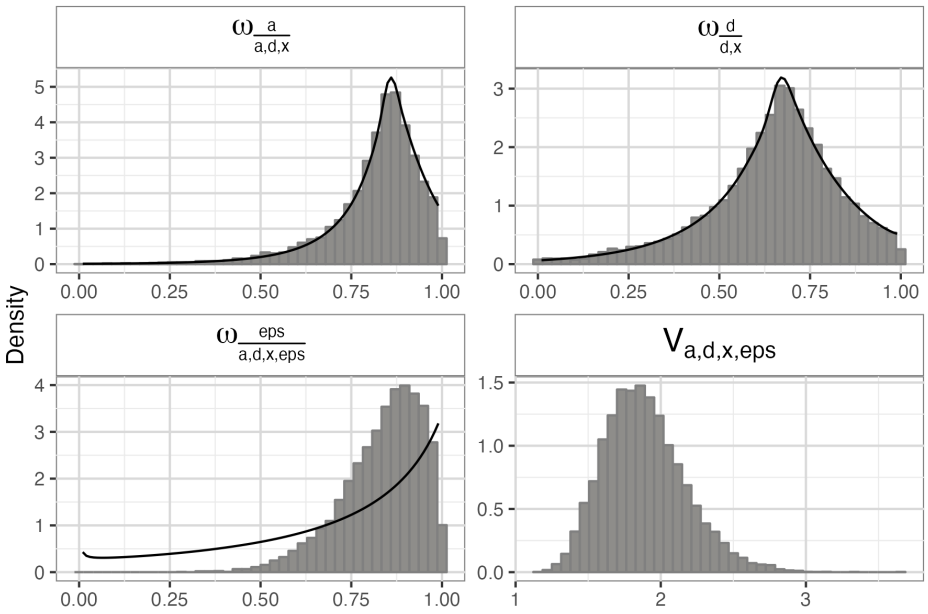


Figure 6: Prior and posterior distribution of the random effect parameters for the genomic selection example.

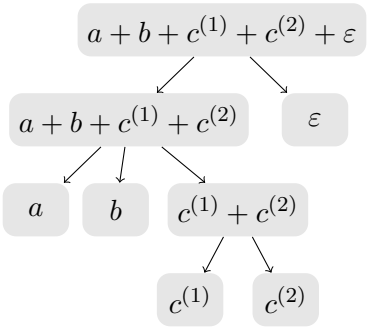
Tree structure	Parameters, priors
	$\sigma^2_{\mathrm{a+b+c^{(1)}+c^{(2)}+\varepsilon}} \sim$ Jeffreys′ $\omega_{\frac{f_i}{f_i+\varepsilon}} \sim \mathrm{PC}_0(0.25)$ $(\omega_{\frac{\mathrm{a}}{f_i}}, \omega_{\frac{\mathrm{b}}{f_i}}, 1-\omega_{\frac{\mathrm{a}}{f_i}}-\omega_{\frac{\mathrm{b}}{f_i}}) \sim \mathrm{Dirichlet}(3)$ $\omega_{\frac{\mathrm{c}^{(1)}}{\mathrm{c}^{(1)}+\mathrm{c}^{(2)}}} \sim \mathrm{PC}_0(0.25)$

Table 3: Tree structures and the corresponding parameters for the prior used in the latin square model: $f_{i,j} = a_i + b_j + c^{(1)}_{k[i,j]} + c^{(2)}_{k[i,j]}$.

Latin square experiment

We consider the latin square experiment in Example 2.2. In line with Fuglstad *et al.* (2020, Section 5.2), we expand the model and assume the treatment effect now consists of a smooth signal $\boldsymbol{c}^{(1)} = (c^{(1)}_1, \ldots, c^{(1)}_9) \sim (\mathbf{0}, \sigma^2_{\mathrm{c}^{(1)}} \mathbf{Q}^{-1}_{\mathrm{RW2}})$ where $\sigma^2_{\mathrm{c}^{(1)}}$ is the variance and $\mathbf{Q}^{-1}_{\mathrm{RW2}}$ is the covariance matrix describing the intrinsic second-order random walk (Rue and Held 2005, Chapter 3), and random noise $\boldsymbol{c}^{(2)} = (c^{(2)}_1, \ldots, c^{(2)}_9) \sim \mathcal{N}_9(\mathbf{0}, \sigma^2_{\mathrm{c}^{(2)}} \mathbf{I}_9)$. We remove implicit intercept and linear effect by requiring $\sum_{i=1}^9 c^{(1)}_i = 0$ and $\sum_{i=1}^9 i c^{(1)}_i = 0$. To simplify the notation, we use $f_{i,j} = a_i + b_j + c^{(1)}_{k[i,j]} + c^{(2)}_{k[i,j]}$.

We show how to create the prior distributions in Table 3. We want to avoid overfitting of the

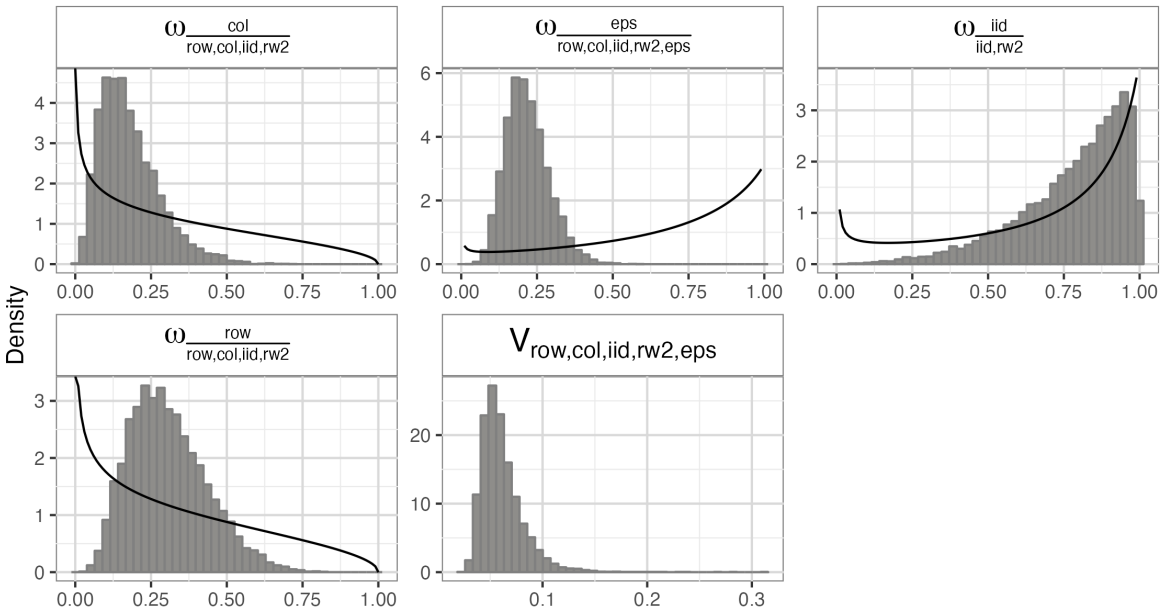


Figure 7: Prior and posterior distribution of the random effect parameters for the latin square example.

model, and use a prior with shrinkage towards the residuals in the top split with a median giving 75% residual effect. We do not have any preference for the attribution of the row, column and treatment effects, and use an ignorant Dirichlet prior for the middle split. In the bottom split we again want to avoid overfitting, and use a prior with shrinkage towards the unstructured treatment effect and a median corresponding to 75% unstructured treatment effect. At last we do not want to say anything about the scale of the total variance, and use the default Jeffreys' prior.

The dataset used in this model is included in **makemyprior** as `latin_data`. It is a simulated dataset, following the description of Fuglstad *et al.* (2020, Section 5.2), where we have used $\sigma_{\text{a}} = \sigma_{\text{b}} = \sigma_{\text{c}(2)} = \sigma_{\varepsilon} = 0.1$ and true treatment effect $c_i^{(1)} = 0.02 \cdot ((i-5)^2 - 20/3)$. `lin` in the formula below is the linear effect of treatment $k[i,j]$ and both the intercept μ and the coefficient β have a default $\mathcal{N}(0, \sigma = 1000)$ prior. The following will fit this model and produce the plots in Figure 7:

```
R> formula <- y ~ lin + mc(row) + mc(col) + mc(iid) +
+    mc(rw2, model = "rw2", constr = TRUE, lin_constr = TRUE)
R> prior <- make_prior(
+    formula, latin_data,
+    prior = list(
+      tree = "s1 = (rw2, iid); s2 = (row, col, s1); s3 = (s2, eps)",
+      w = list(
+        s1 = list(prior = "pc0", param = 0.25),
+        s2 = list(prior = "dirichlet"),
+        s3 = list(prior = "pc0", param = 0.25))))
```

```
R> posterior <- inference_stan(prior, iter = 15000, warmup = 5000,
+    seed = 1, init = "0", chains = 1,
+    control = list(adapt_delta = 0.9))
R> plot_posterior_stan(posterior, param = "prior", prior = TRUE)

Tree structure: iid_rw2 = (iid,rw2); row_col_iid_rw2 = (row,col,iid_rw2);
eps_row_col_iid_rw2 = (eps,row_col_iid_rw2)

Weight priors:
        w[iid/iid_rw2] ~ PC1(0.75)
        (w[row/row_col_iid_rw2], w[col/row_col_iid_rw2]) ~ Dirichlet(3)
        w[eps/eps_row_col_iid_rw2] ~ PC1(0.75)
Total variance priors:
        V[eps_row_col_iid_rw2] ~ Jeffreys'

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000158 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition
would take 1.58 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:     1 / 15000 [  0%]  (Warmup)
Chain 1: Iteration:  1500 / 15000 [ 10%]  (Warmup)
Chain 1: Iteration:  3000 / 15000 [ 20%]  (Warmup)
Chain 1: Iteration:  4500 / 15000 [ 30%]  (Warmup)
Chain 1: Iteration:  5001 / 15000 [ 33%]  (Sampling)
Chain 1: Iteration:  6500 / 15000 [ 43%]  (Sampling)
Chain 1: Iteration:  8000 / 15000 [ 53%]  (Sampling)
Chain 1: Iteration:  9500 / 15000 [ 63%]  (Sampling)
Chain 1: Iteration: 11000 / 15000 [ 73%]  (Sampling)
Chain 1: Iteration: 12500 / 15000 [ 83%]  (Sampling)
Chain 1: Iteration: 14000 / 15000 [ 93%]  (Sampling)
Chain 1: Iteration: 15000 / 15000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 12.988 seconds (Warm-up)
Chain 1:                23.347 seconds (Sampling)
Chain 1:                36.335 seconds (Total)
Chain 1:
```

Figure 7 shows the prior and posterior together on the parameterization of the prior. The posterior distribution of the bottom split, attributing the treatment effect to the random noise and smooth signal, is only slightly different from the prior, indicating that there is no strong signal about the smooth treatment effect in the data. By using a prior with shrinkage towards only random noise treatment effect, we avoid overfitting. The model has learned about the three other variance proportions, and we see that even though the prior on the amount of

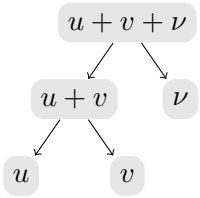
Tree structure	Parameters, priors
	$\sigma^2_{\text{u+v+}\nu} \sim \text{PC}_0(3.35, 0.05)$ $\omega_{\frac{\text{u+v}}{\text{u+v+}\nu}} \sim \text{PC}_1(0.75)$ $\omega_{\frac{\text{u}}{\text{u+v}}} \sim \text{PC}_0(0.25)$

Table 4: Tree structures and the corresponding parameters for the neonatal mortality model.

total variance going to the residual effect has shrinkage towards 1, the model is not restricted by this (top right plot).

Note that for inference with **INLA**, we remove implicit linear effect by adding `extraconstr = list(A = matrix(1:9, 1, 9), e = matrix(0, 1, 1))` to `mc(rw2, ...)` in the formula.

6.2. Binomial responses: Neonatal mortality

This example is based on a study carried out by Fuglstad *et al.* (2020). Child mortality is an important indicator of health and well-being in a country. We define neonatal mortality as the number of deaths of infants the first month of life per live birth, which can be estimated using national household surveys from Demographic and Health Surveys (Kenya National Bureau of Statistics *et al.* 2015). From such surveys we can extract the number of live births $b_{i,j}$ and the number of neonatal deaths $y_{i,j}$ in cluster j in county i, and use an indicator $x_{i,j}$ for classifying cluster j in county i as rural ($x_{i,j} = 0$) or urban (1). We model $y_{i,j}|b_{i,j}, p_{i,j} \sim$ Binomial$(b_{i,j}, p_{i,j})$ with the linear predictor

$$\eta_{i,j} = \text{logit}(p_{i,j}) = \mu + x_{i,j}\beta + u_i + v_i + \nu_{i,j}, \;\; i = 1, \ldots, n, \;\; j = 1, \ldots, m_i, \tag{4}$$

where $v_i \sim \mathcal{N}(0, \sigma_v^2)$ and $\nu_{i,j} \sim \mathcal{N}(0, \sigma_\nu^2)$ are i.i.d. random effects with sum-to-zero constraints for counties and clusters, respectively, and $\boldsymbol{u}$ is a Besag effect on county with variance σ_u^2 and a sum-to-zero constraint. In the Besag model, the spatial effect of each county depends on the effects in the neighboring regions (see e.g., Besag, York, and Mollié (1991) for details), and when combining it with an i.i.d. effect on the same level in the hierarchy, we get a BYM (Besag, York, and Mollié) model (Besag *et al.* 1991). We want to investigate whether or not there is a spatial effect present.

We simulated a dataset following the description in Fuglstad *et al.* (2020, Section 6.2) with the 47 counties in Kenya. We used 6, 7 or 8 clusters in each county which gave in total 327 clusters, and thus 327 observations, $b_{i,j} = 25$ live births in each cluster, and parameters $\mu = -4$, $\beta = 0.1$, $\sigma_\nu^2 = 0.2$, $\sigma_{\text{v}}^2 = 0.1$, and $\sigma_{\text{u}}^2 = 0.5$.

This dataset is available in **makemyprior** as `neonatal_data`, together with other necessary files for fitting the model.

We prefer coarser over finer unstructured effects, and unstructured over structured effects. That means that we prefer $\boldsymbol{v}$ over $\boldsymbol{u}$ and $\boldsymbol{v} + \boldsymbol{u}$ over $\boldsymbol{\nu}$ in the prior. The BYM model is intuitively represented with a dual split in the prior tree, where one leaf node represents a Besag effect and the other represents an i.i.d. effect. We achieve this with a prior that distributes the between-county variance with shrinkage towards the unstructured county effect, which

gives the BYM2 model of Riebler *et al.* (2016), and we shrink the total variance towards the county effects. Following Fuglstad *et al.* (2020), we induce shrinkage on the total variance such that we have a 90% credible interval of $(0.1, 10)$ for the effect of $\exp(v_i + u_i + \nu_{i,j})$. We use the function `find_pc_prior_param()` in **makemyprior** to find the parameters for the PC prior:

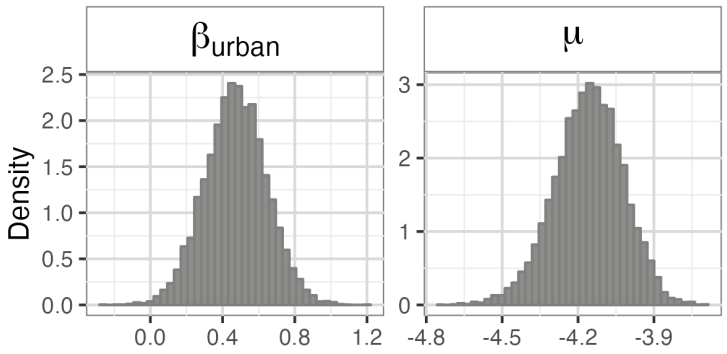
```
R> set.seed(1)
R> find_pc_prior_param(lower = 0.1, upper = 10, prob = 0.9, N = 2e5)

U = 3.353132
Prob(0.09866969 < exp(eta) < 9.892902) = 0.9
```

This gives a $\text{PC}_0(3.35, 0.05)$ prior. The tree structure and a summary of the prior distributions can be found in Table 4. We fit the model with Stan:

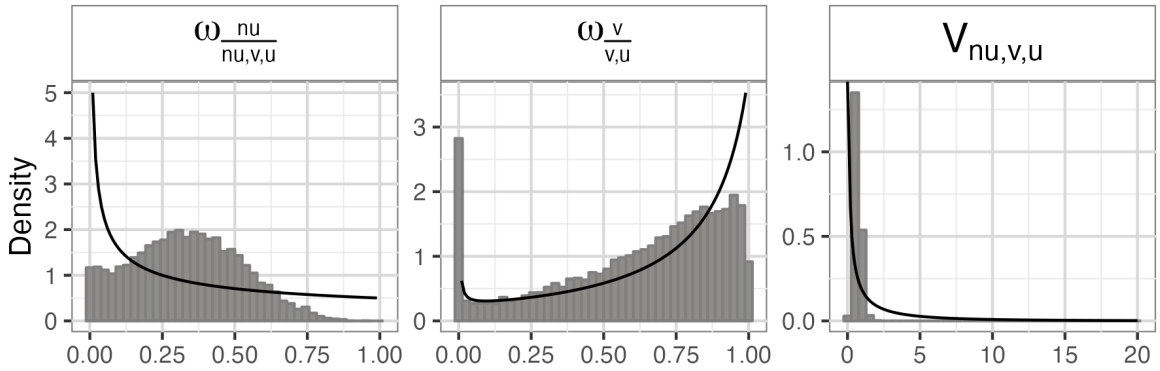
```
R> graph_path <- paste0(path.package("makemyprior"), "/neonatal.graph")
R> formula <- y ~ mc(nu) + mc(v) +
+    mc(u, model = "besag", graph = graph_path, scale.model = TRUE)
R> prior <- make_prior(
+    formula, neonatal_data, family = "binomial",
+    prior = list(
+      tree = "s1 = (u, v); s2 = (s1, nu)",
+      w = list(
+        s1 = list(prior = "pc0", param = 0.25),
+        s2 = list(prior = "pc1", param = 0.75)),
+      V = list(s2 = list(prior = "pc", param = c(3.35, 0.05)))))
R> posterior <- inference_stan(prior, iter = 15000, warmup = 5000,
+   seed = 1, init = "0", chains = 1, control = list(adapt_delta = 0.95))

Tree structure: v_u = (v,u); nu_v_u = (nu,v_u)

Weight priors:
        w[v/v_u] ~ PC1(0.75)
        w[nu/nu_v_u] ~ PC0(0.25)
Total variance priors:
        sqrt(V)[nu_v_u] ~ PC0(3.35, 0.05)

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000169 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition
would take 1.69 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:     1 / 15000 [  0%]  (Warmup)
Chain 1: Iteration:  1500 / 15000 [ 10%]  (Warmup)
```

```
Chain 1: Iteration:  3000 / 15000 [ 20%]  (Warmup)
Chain 1: Iteration:  4500 / 15000 [ 30%]  (Warmup)
Chain 1: Iteration:  5001 / 15000 [ 33%]  (Sampling)
Chain 1: Iteration:  6500 / 15000 [ 43%]  (Sampling)
Chain 1: Iteration:  8000 / 15000 [ 53%]  (Sampling)
Chain 1: Iteration:  9500 / 15000 [ 63%]  (Sampling)
Chain 1: Iteration: 11000 / 15000 [ 73%]  (Sampling)
Chain 1: Iteration: 12500 / 15000 [ 83%]  (Sampling)
Chain 1: Iteration: 14000 / 15000 [ 93%]  (Sampling)
Chain 1: Iteration: 15000 / 15000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 18.781 seconds (Warm-up)
Chain 1:                74.789 seconds (Sampling)
Chain 1:                93.57 seconds (Total)
Chain 1:
```

For inference with **INLA**, the `Ntrials` argument must be provided to `inference_inla()`. The following produce the plots in Figure 8 and gives some key information on the posterior:



(a) Posterior of the effect of urban/rural and the intercept.



(b) Random effect parameters.

Figure 8: Prior and posterior distribution of (a) coefficients of the fixed effects and (b) total variance and variance proportions of the random effects for the neonatal mortality example.

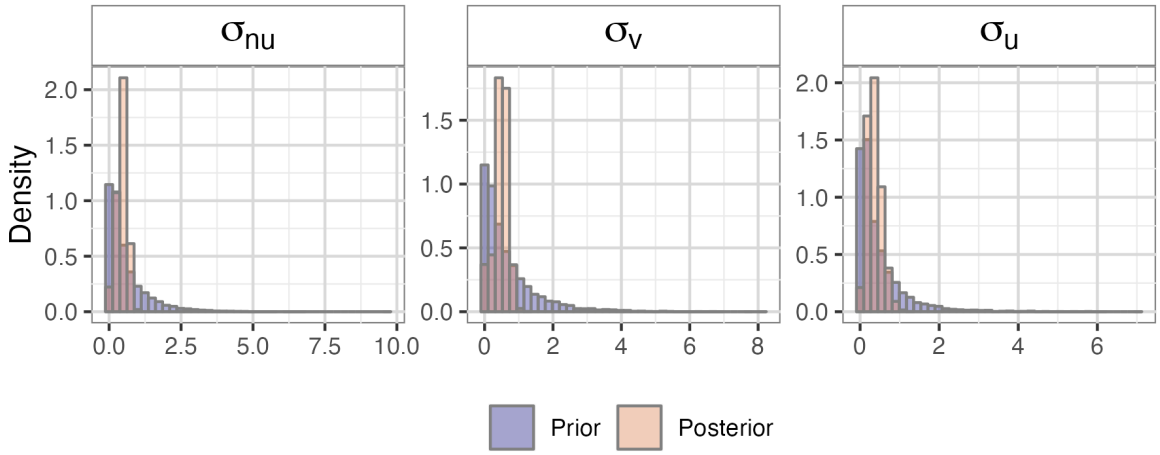


Figure 9: Prior and posterior distribution for the neonatal mortality example on standard deviation scale.

```
R> plot_posterior_fixed(posterior)
R> plot_posterior_stan(posterior, param = "prior", prior = TRUE)
R> posterior

Model: y ~ urban + mc(nu) + mc(v) + mc(u, model = "besag", graph = graph_path,
    scale.model = TRUE)
Tree structure: v_u = (v,u); nu_v_u = (nu,v_u)

Inference done with Stan.

 Param.       mean   median sd
 V[nu_v_u]     0.667  0.637 0.245
 w[v/v_u]      0.620  0.696 0.294
 w[nu/nu_v_u]  0.333  0.329 0.187
 intercept    -4.159 -4.154 0.136
 urban         0.473  0.473 0.170
```

The samples for the posterior spatial effects u_i can easily be extracted with the function `extract_posterior_effect()` which take the arguments `obj` from `inference_stan()` and the name of the effect:

```
R> u <- extract_posterior_effect(posterior, "u")
```

The fixed effects in Figure 8a show that the intercept is not contributing much to the linear predictor, and that there is a higher mortality in urban areas (which is the case also for real data, see Kenya National Bureau of Statistics *et al.* 2015). From Figure 8b we see that the model has learned about the total variance from the data and about the amount of total (latent) variance to the cluster effect ($\boldsymbol{\nu}$), but there is not enough information in the data about the amount of county variance to the structured county effect ($\boldsymbol{u}$). The following fits

the model without the likelihood (sampling from the prior) and produces the plots of the prior and posterior on standard deviation scale in Figure 9:

```
R> prior_samps <- inference_stan(prior, use_likelihood = F, print_prior = F,
+    iter = 15000, warmup = 5000,
+    seed = 1, init = "0", chains = 1)
R> plot_several_posterior_stan(list(Prior = prior_samps,
+    Posterior = posterior), "stdev")
```

```
SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000134 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition
would take 1.34 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:     1 / 15000 [  0%]  (Warmup)
Chain 1: Iteration:  1500 / 15000 [ 10%]  (Warmup)
Chain 1: Iteration:  3000 / 15000 [ 20%]  (Warmup)
Chain 1: Iteration:  4500 / 15000 [ 30%]  (Warmup)
Chain 1: Iteration:  5001 / 15000 [ 33%]  (Sampling)
Chain 1: Iteration:  6500 / 15000 [ 43%]  (Sampling)
Chain 1: Iteration:  8000 / 15000 [ 53%]  (Sampling)
Chain 1: Iteration:  9500 / 15000 [ 63%]  (Sampling)
Chain 1: Iteration: 11000 / 15000 [ 73%]  (Sampling)
Chain 1: Iteration: 12500 / 15000 [ 83%]  (Sampling)
Chain 1: Iteration: 14000 / 15000 [ 93%]  (Sampling)
Chain 1: Iteration: 15000 / 15000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 2.402 seconds (Warm-up)
Chain 1:                15.269 seconds (Sampling)
Chain 1:                17.671 seconds (Total)
Chain 1:
```

From these graphs we see that the posterior of the standard deviations are clearly different from the prior. We saw in Figure 8b that the model did not learn much about the amount of county variation accounted for by the Besag effect ($\boldsymbol{u}$), but we cannot see this from plots of the posterior standard deviations, and they do not show the whole picture. This is another advantage of the HD prior: it is easy to see that even though we get the impression that the model has learned from the data, that knowledge is not necessarily about the whole model. This shows, as Fuglstad *et al.* (2020) points out, that one should be careful before drawing conclusions on first impressions about the results, and more investigation should be done.

7. Summary and discussion

The **makemyprior** package offers an intuitive and transparent way of choosing and visualizing prior distributions for Bayesian hierarchical models. It is easy to utilize expert knowledge,

and clear what prior distributions are used. The package works with the flexible and widely used latent Gaussian models, and offers Gaussian, binomial and Poisson likelihoods. This, together with the latent models included, makes the package applicable in a range of problems and applications.

The package can be used to investigate the prior choices, and the user can simulate from the prior with Stan and look at the prior distributions on different parameterizations. In this way, crucial misunderstandings of what prior distributions are used can be discovered and corrected, and thus increase the understanding and meaning of the prior. The usage of the hierarchical decomposition (HD) prior is not limited to inference carried out with **rstan** or **INLA**. Joint priors can be constructed and evaluated using **makemyprior** for use in external inference code or potentially incorporated in other R packages.

To see how the individual fixed effects contribute to the total data variation would be interesting, and could be done with the HD prior framework. However, fixed effects are often correlated, and the variance that is explained by each single fixed effect is not well defined. The perhaps most intuitive way to include fixed effects directly in the tree structure is to assign one variance parameter to each effect, but this can quickly increase the amount of variance parameters to a level where inference become computationally hard. Gelman and Hill (2007) and Zhang, Naughton, Bondell, and Reich (2022) have proposed prior distributions related to the coefficient of determination, R^2, which measures the amount of variance explained by the model. The generalized R^2 proposed by Gelman and Hill (2007) measures this at each level in the hierarchical model. Yanchenko, Bondell, and Reich (2021) extended the framework by Zhang *et al.* (2022) to generalized linear mixed models. Inclusion of fixed effects directly in the joint prior in the HD prior framework is discussed by Fuglstad *et al.* (2020). We have chosen to give them independent and vague priors.

Other exiting additions to the HD prior framework include extending it to models outside the class of latent Gaussian models, or to models where the hyperparameters of the priors will get prior distributions. This will require further development of the framework, and may be highly computationally expensive, as the penalized complexity (PC) prior cannot be pre-computed in the same way as we do now with the conditioning on the hyperparameters.

A natural first extension of the functionality of **makemyprior** and also the HD prior framework might be to support Dirichlet distributions with custom hyperparameter values for the variance proportions, and custom defined beta priors in the case of a dual split. This addition to the framework will open for even easier integration of the HD prior in other software (such as Template Model Builder **TMB**, Kristensen *et al.* 2016) as expert knowledge can be included without having to compute the PC prior. However, this will complicate the intuition behind the prior, and it will be more difficult to use prior and expert knowledge in a transparent way. It will require more thoughtful prior choices, and we lose one of the advantages with the easy-to-use and intuitive way of making priors with **makemyprior** which now are guaranteed through the properties (such as shrinkage) of the PC prior. The variance prior distributions included in the package cover the most popular choices, but allowing the user to specify custom prior distributions for variance parameters will further increase the flexibility of the package.

Including more latent models will further increase the amount of applications the package can be used for without specifying custom Stan code. This includes handling parameters such as correlations, which can be done by giving them independent priors and conditioning on a

representative value (e.g., the mean or median of the chosen prior) for this parameter when the joint prior is computed. For prior elicitation of these parameters additional R packages might be useful. For example, the R package **meta4diag** (Guo and Riebler 2018), which considers the analysis of diagnostic test studies, offers three strategies to intuitively define a penalized complexity prior for a correlation parameter ρ in a bivariate model given an arbitrary base value ρ_0, while the **INLA** package implements PC priors for autoregressive models (Sørbye and Rue 2017), see functions `pc.cor0` and `pc.cor1`. To open for easy integration into other software for inference, such as **TMB**, can be useful for models that are very complex and will be highly time consuming and difficult to fit with **rstan** or **INLA**.

In conclusion, **makemyprior** offers something not offered by the range of packages that can be used to carry out inference for Bayesian hierarchical models. It makes it easy to include prior knowledge in an intuitive and transparent way, can be used to verify prior choices, and allows direct inference in a simple way. **makemyprior** makes users aware of what priors are used and makes prior selection a conscious choice, which is important when doing inference to ensure that the model fitted is indeed the intended one.

Computational details

The results in this paper were obtained with R version 4.4.0 on platform aarch64-apple-darwin20 (64-bit) running under macOS Sonoma 14.1.1.

Package versions: **makemyprior** 1.2.2, **ggplot2** 3.5.1, **Matrix** 1.7-0, **methods** 4.4.0, **shiny** 1.8.1.1, **shinyjs** 2.1.0, **shinyBS** 0.61.1, **visNetwork** 2.1.2, **rlang** 1.1.3, **MASS** 7.3-60.2, **rstan** 2.32.6, **INLA** 24.5.10 **knitr** 1.47, and **rmarkdown** 2.27.

Acknowledgments

Hem, Fuglstad and Riebler were supported by project number 240873 from the Research Council of Norway.

References

Banerjee S, Carlin BP, Gelfand AE (2014). *Hierarchical Modeling and Analysis for Spatial Data*. Chapman & Hall/CRC, Boca Raton.

Berger JO, Bernardo JM, Sun D (2009). "The Formal Definition of Reference Priors." *The Annals of Statistics*, **37**(2), 905–938. doi:10.1214/07-AOS587.

Besag J, York J, Mollié A (1991). "Bayesian Image Restoration, with Two Applications in Spatial Statistics." *Annals of the Institute of Statistical Mathematics*, **43**(1), 1–20. doi:10.1007/BF00116466.

Bürkner PC (2017). "**brms**: An R Package for Bayesian Multilevel Models Using Stan." *Journal of Statistical Software*, **80**, 1–28. doi:10.18637/jss.v080.i01.

Caldwell A, Kollár D, Kröninger K (2009). "**BAT** – The Bayesian Analysis Toolkit." *Computer Physics Communications*, **180**(11), 2197–2209. doi:10.1016/j.cpc.2009.06.026.

Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "Stan: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1). doi:10.18637/jss.v076.i01.

Chang W, Cheng J, Allaire JJ, Sievert C, Schloerke B, Xie Y, Allen J, McPherson J, Dipert A, Borges B (2024). ***shiny**: Web Application Framework for R*. doi:10.32614/CRAN.package.shiny. R package version 1.9.1.

Cressie N, Wikle CK (2011). *Statistics for Spatio-Temporal Data*. John Wiley & Sons, Hoboken.

Depaoli S, Winter SD, Visser M (2020). "The Importance of Prior Sensitivity Analysis in Bayesian Statistics: Demonstrations Using an Interactive **shiny** App." *Frontiers in Psychology*, **11**, 608045. doi:10.3389/fpsyg.2020.608045.

Faux AM, Gorjanc G, Gaynor RC, Battagin M, Edwards SM, Wilson DL, Hearne SJ, Gonen S, Hickey JM (2016). "**AlphaSim**: Software for Breeding Program Simulation." *The Plant Genome*, **9**(3), 1–14. doi:10.3835/plantgenome2016.02.0013.

Fong Y, Rue H, Wakefield J (2010). "Bayesian Inference for Generalized Linear Mixed Models." *Biostatistics*, **11**(3), 397–412. ISSN 1465-4644. doi:10.1093/biostatistics/kxp053.

Franco-Villoria M, Ventrucci M, Rue H (2020). "Variance Partitioning in Spatio-Temporal Disease Mapping Models." *arXiv 2109.13374*, arXiv.org E-Print Archive. doi:10.48550/arXiv.2109.13374.

Fuglstad GA, Hem IG, Knight A, Rue H, Riebler A (2020). "Intuitive Joint Priors for Variance Parameters." *Bayesian Analysis*, **15**(4), 1109–1137. doi:10.1214/19-BA1185.

Gabry J, Veen D (2022). ***shinystan**: Interactive Visual and Numerical Diagnostics and Posterior Analysis for Bayesian Models*. doi:10.32614/CRAN.package.shinystan. R package version 2.6.0.

Gaynor C (2023). ***AlphaSimR**: Breeding Program Simulations*. doi:10.32614/CRAN.package.AlphaSimR. R package version 1.5.3.

Gaynor RC, Gorjanc G, Bentley AR, Ober ES, Howell P, Jackson R, Mackay IJ, Hickey JM (2017). "A Two-Part Strategy for Using Genomic Selection to Develop Inbred Lines." *Crop Science*, **57**(5), 2372–2386. doi:10.2135/cropsci2016.09.0742.

Gelman A, Carlin JB, Stern HS, Dunson DB, Vehtari A, Rubin DB (2013). *Bayesian Data Analysis*. Chapman & Hall/CRC, Boca Raton.

Gelman A, Hill J (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*, volume 1. Cambridge University Press, New York.

Gelman A, Simpson D, Betancourt M (2017). "The Prior Can Often Only Be Understood in the Context of the Likelihood." *Entropy*, **19**(10), 555. doi:10.3390/e19100555.

Gelman A, Sturtz S, Ligges U, Gorjanc G, Kerman J, Thomas N (2020a). ***R2OpenBUGS**: Running **OpenBUGS** from R*. doi:10.32614/CRAN.package.R2OpenBUGS. R package version 3.2-3.2.1.

Gelman A, Vehtari A, Simpson D, Margossian CC, Carpenter B, Yao Y, Kennedy L, Gabry J, Bürkner PC, Modrák M (2020b). "Bayesian Workflow." *arXiv 2011.01808*, arXiv.org E-Print Archive. doi:10.48550/arXiv.2011.01808.

Goel PK, Degroot MH (1981). "Information About Hyperparameters in Hierarchical Models." *Journal of the American Statistical Association*, **76**(373), 140–147. doi:10.1080/01621459.1981.10477620.

Goodrich B, Gabry J, Ali I, Brilleman S (2020). ***rstanarm**: Bayesian Applied Regression Modeling via Stan*. R package version 2.21.1, URL https://mc-stan.org/rstanarm.

Guo J, Riebler A (2018). "**meta4diag**: Bayesian Bivariate Meta-Analysis of Diagnostic Test Studies for Routine Practice." *Journal of Statistical Software*, **83**, 1–31. doi:10.18637/jss.v083.i01.

Guo J, Riebler A, Rue H (2017). "Bayesian Bivariate Meta-Analysis of Diagnostic Test Studies with Interpretable Priors." *Statistics in Medicine*, **36**(19), 3039–3058. doi:10.1002/sim.7313.

Hem IG, Selle M, Gorjanc G, Fuglstad GA, Riebler A (2020). "Supplemental Material for "Robust Modelling of Additive and Non-Additive Variation with Intuitive Inclusion of Expert Knowledge"." doi:10.6084/m9.figshare.12040716.

Hem IG, Selle ML, Gorjanc G, Fuglstad GA, Riebler A (2021). "Robust Modeling of Additive and Nonadditive Variation with Intuitive Inclusion of Expert Knowledge." *Genetics*, **217**(3). doi:10.1093/genetics/iyab002. iyab002.

Hoffman MD, Gelman A (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research*, **15**(1), 1593–1623. URL http://jmlr.org/papers/v15/hoffman14a.html.

Holand AM, Steinsland I, Martino S, Jensen H (2013). "Animal Models and Integrated Nested Laplace Approximations." *G3: Genes, Genomes, Genetics*, **3**(8), 1241–1251. ISSN 2160-1836. doi:10.1534/g3.113.006700.

Jeffreys H (1946). "An Invariant Form for the Prior Probability in Estimation Problems." *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, **186**(1007), 453–461. doi:10.1098/rspa.1946.0056.

Kenya National Bureau of Statistics, Ministry of Health/Kenya, National AIDS Control Council/Kenya, Kenya Medical Research Institute, National Council for Population and Development/Kenya (2015). "Kenya Demographic and Health Survey 2014." URL https://dhsprogram.com/pubs/pdf/FR308/FR308.pdf.

Kristensen K, Nielsen A, Berg CW, Skaug H, Bell BM (2016). "**TMB**: Automatic Differentiation and Laplace Approximation." *Journal of Statistical Software*, **70**(5), 1–21. doi:10.18637/jss.v070.i05.

Lambert PC, Sutton AJ, Burton PR, Abrams KR, Jones DR (2005). "How Vague Is Vague? A Simulation Study of the Impact of the Use of Vague Prior Distributions in MCMC Using **WinBUGS**." *Statistics in Medicine*, **24**(15), 2401–2428. doi:10.1002/sim.2112.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "**WinBUGS** – A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**(4), 325–337. doi:10.1023/A:1008929526011.

Plummer M (2017). ***JAGS** Version 4.3.0 User Manual*. URL https://sourceforge.net/projects/mcmc-jags/files/Manuals/4.x.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Riebler A, Sørbye SH, Simpson D, Rue H (2016). "An Intuitive Bayesian Spatial Model for Disease Mapping That Accounts for Scaling." *Statistical Methods in Medical Research*, **25**(4), 1145–1165. doi:10.1177/0962280216660421.

Rue H, Held L (2005). *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall/CRC, Boca Raton. doi:10.1201/9780203492024.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian Inference for Latent Gaussian Models by Using Integrated Nested Laplace Approximations." *Journal of the Royal Statistical Society B*, **71**(2), 319–392. doi:10.1111/j.1467-9868.2008.00700.x.

Rue H, Riebler A, Sørbye SH, Illian JB, Simpson DP, Lindgren FK (2017). "Bayesian Computing with INLA: A Review." *Annual Review of Statistics and Its Application*, **4**, 395–421. doi:10.1146/annurev-statistics-060116-054045.

Selle ML, Steinsland I, Hickey JM, Gorjanc G (2019). "Flexible Modelling of Spatial Variation in Agricultural Field Trials with the R Package **INLA**." *Theoretical and Applied Genetics*, **132**(12), 3277–3293. doi:10.1007/s00122-019-03424-y.

Simpson D, Rue H, Riebler A, Martins TG, Sørbye SH (2017). "Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors." *Statistical Science*, **32**(1), 1–28. doi:10.1214/16-STS576.

Slater JJ, Brown PE, Rosenthal JS, Mateu J (2022). "Capturing Spatial Dependence of COVID-19 Case Counts with Cellphone Mobility Data." *Spatial Statistics*, **49**, 100540. doi:10.1016/j.spasta.2021.100540.

Smid SC, Winter SD (2020). "Dangers of the Defaults: A Tutorial on the Impact of Default Priors When Using Bayesian SEM With Small Samples." *Frontiers in Psychology*, **11**, 3536. doi:10.3389/fpsyg.2020.611963.

Sørbye SH, Rue H (2014). "Scaling Intrinsic Gaussian Markov Random Field Priors in Spatial Modelling." *Spatial Statistics*, **8**, 39–51. doi:10.1016/j.spasta.2013.06.004.

Sørbye SH, Rue H (2017). "Penalised Complexity Priors for Stationary Autoregressive Processes." *Journal of Time Series Analysis*, **38**(6), 923–935. doi:10.1111/jtsa.12242.

Sorensen D, Gianola D (2007). *Likelihood, Bayesian, and MCMC Methods in Quantitative Genetics*. Springer-Verlag, New York.

Spiegelhalter D, Thomas A, Best N, Lunn D (2014). ***OpenBUGS** User Manual*. Version 3.2.3.

Stan Development Team (2024). ***rstan**: The R Interface to Stan*. R package version 2.21.2, URL https://mc-stan.org/.

Sturtz S, Ligges U, Gelman A (2005). "**R2WinBUGS**: A Package for Running **WinBUGS** from R." *Journal of Statistical Software*, **12**, 1–16. doi:10.18637/jss.v012.i03.

Vehtari A, Gabry J, Magnusson M, Yao Y, Bürkner PC, Paananen T, Gelman A (2020). ***loo**: Efficient Leave-One-Out Cross-Validation and WAIC for Bayesian Models*. R package version 2.4.1, URL https://mc-stan.org/loo/.

Wakefield J (2006). "Disease Mapping and Spatial Regression with Count Data." *Biostatistics*, **8**(2), 158–183. ISSN 1465-4644. doi:10.1093/biostatistics/kxl008.

Yanchenko E, Bondell HD, Reich BJ (2021). "The R2D2 Prior for Generalized Linear Mixed Models." *arXiv 2111.10718*, arXiv.org E-Print Archive. doi:10.48550/arXiv.2111.10718.

Zhang YD, Naughton BP, Bondell HD, Reich BJ (2022). "Bayesian Regression Using a Prior on the Model Fit: The R2-D2 Shrinkage Prior." *Journal of the American Statistical Association*, **117**(538), 862–874. doi:10.1080/01621459.2020.1825449.

Zondervan-Zwijnenburg M, Peeters M, Depaoli S, Van de Schoot R (2017). "Where Do Priors Come From? Applying Guidelines to Construct Informative Priors in Small Sample Research." *Research in Human Development*, **14**(4), 305–320. doi:10.1080/15427609.2017.1370966.

Affiliation:

Ingeborg Hem
Department of Mathematical Sciences
Norwegian University of Science and Technology
Alfred Getz' vei 1, 7034 Trondheim, Norway
E-mail: ingeborg.hem@ntnu.no

Journal of Statistical Software https://www.jstatsoft.org/
published by the Foundation for Open Access Statistics https://www.foastat.org/
August 2024, Volume 110, Issue 3 *Submitted:* 2022-02-24
doi:10.18637/jss.v110.i03 *Accepted:* 2024-05-13