shinybrms: Fitting Bayesian Regression Models Using a Graphical User Interface for the R Package brms

by Frank Weber, Katja Ickstadt, and Änne Glass

Abstract Despite their advantages, the application of Bayesian regression models is still the exception compared to frequentist regression models. Here, we present our R package **shinybrms** which provides a graphical user interface for fitting Bayesian regression models, with the frontend consisting of a **shiny** app and the backend relying on the R package **brms** which in turn relies on Stan. With **shinybrms**, we hope that Bayesian regression models (and regression models in general) will become more popular in applied research, data analyses, and teaching. Here, we illustrate our graphical user interface by the help of an example from medical research.

1 Introduction

The relevance of regression models in applied research has already been well pointed out, for example, by Karabatsos (2015):

> "Regression modeling is ubiquitous in empirical areas of scientific research. This is because most research questions can be asked in terms of how a dependent variable changes as a function of one or more covariates (predictors)."

Conducting regression analyses in a *Bayesian* framework has a lot of advantages. Introductory texts on Bayesian statistics (in general) are, e.g., McElreath (2020), Albert and Hu (2019), Reich and Ghosh (2019), StataCorp (2019a), Gelman et al. (2020a), and Johnson et al. (2022). For readers with little background in Bayesian statistics, we recommend reading one of these textbooks first. A more detailed introduction may be found, e.g., in Gelman et al. (2014). In particular, Bayesian statistics has the following advantages (for further advantages, see, e.g., Gelman et al., 2014; StataCorp, 2019a):

1. Bayesian methods allow for incorporation of *prior knowledge*. Generally, inclusion of prior knowledge is desirable: The flat prior implied by the frequentist maximum likelihood (ML) method may lead to nonsensical inferences (Gelman et al., 2017). Even if the inclusion of informative prior knowledge is not desired, weakly informative priors have the advantage (compared to so-called "noninformative" priors[1]) to downweight unreasonable parameter values and to introduce a certain *regularization* or *penalization*, helping against overfitting (Gelman, 2006; Gelman et al., 2008, 2014, 2017). Hereafter, we follow conventional notation in Bayesian statistics and denote the prior for the parameter vector $\boldsymbol{\theta}$ by $p(\boldsymbol{\theta})$.
2. Similarly, the prior distribution may be used to impose *parameter constraints* in an easy and natural way. There is no need for *ad-hoc* solutions to cut off parameter estimates. For example, many frequentist between-study variance estimators in the random-effects meta-analysis model are cut off at zero.
3. It is usually possible to infer the posterior *exactly* (apart from minor approximations such as those arising from the Monte Carlo error). In that case, Bayesian statistics does not need to resort to large-sample approximations such as the asymptotic normal distribution of the ML estimator often used in frequentist statistics.
4. For most practical cases, Markov chain Monte Carlo (MCMC) sampling (see section Markov chain Monte Carlo) constitutes a *generic* Bayesian inference method. In frequentist statistics, generic methods such as the asymptotic normal distribution of the ML estimator can be unsatisfactory, e.g., for small sample sizes. This is why in frequentist statistics, different inferential methods have often evolved for the same task or model. This complicates frequentist analyses for users, especially for those with little background in statistics.
5. The quantities derived from the posterior have a *more intuitive interpretation* than their frequentist counterparts which are based on the sampling distribution of the estimator. In particular, Bayesian posterior intervals (credible intervals, CrIs) have the interpretation that is often incorrectly attributed to frequentist confidence intervals (CIs) (McElreath, 2020) and posterior tail-area probabilities have the interpretation that is often incorrectly attributed to frequentist p-values.

[1]We added quotation marks here since noninformative priors might be more informative than intended (Gelman et al., 2017).

6. In Bayesian statistics, *uncertainty in nuisance parameters* is easily—and naturally—taken into account by integrating them out from the posterior:

$$p(\boldsymbol{\psi}|\boldsymbol{\mathcal{D}}) = \int p(\boldsymbol{\psi}, \boldsymbol{\phi}|\boldsymbol{\mathcal{D}})\, \mathrm{d}\boldsymbol{\phi} \tag{1}$$

with $\boldsymbol{\mathcal{D}}$ denoting the data, $\boldsymbol{\psi}$ the parameter vector of interest, and $\boldsymbol{\phi}$ the nuisance parameter vector (so that $(\boldsymbol{\psi}^\intercal, \boldsymbol{\phi}^\intercal)^\intercal = \boldsymbol{\theta}$)[2]. Taking uncertainty in nuisance parameters into account helps against overfitting (like the penalization mentioned in enumeration point number 1 above), but in general, this is not that easy in frequentist statistics, as can be seen in random-effects meta-analyses (Weber et al., 2021).

7. When combined with probabilistic programming (as done, e.g., by the various sampling methods introduced in section Algorithms for inferring the posterior), a Bayesian analysis naturally *propagates the posterior uncertainty* into derived quantities (Gelman et al., 2020a).

8. Often, frequentist analyses result in the typical *null-hypothesis significance testing* which is being criticized to an increasing degree (Amrhein and Greenland, 2018; Amrhein et al., 2019; McShane et al., 2019). Null-hypothesis significance testing is especially problematic for null hypotheses consisting of only a point in parameter space. Of course, Bayesian analyses may also result in null-hypothesis significance testing or similar hypothesis-testing procedures, but in our experience, this is not as common as in frequentist analyses. We designed our software presented here in a way that does not encourage null-hypothesis significance testing.

9. *Posterior predictive checks (PPCs)*—which should be part of a Bayesian workflow (Gelman et al., 2020b)—are an easy and intuitive way of performing model diagnostics in a Bayesian framework, even though the choice and interpretation of the PPCs require some experience (Gelman et al., 2020b). In a frequentist framework, model diagnostics are often not that easy to perform, at least if the uncertainty from parameter estimation should be taken into account.

These advantages will be illustrated in the context of the example from section Example, by comparing our Bayesian analysis to a frequentist one (see section "Frequentist analysis of the example" in the online Supplement file 'Supplement_sections.pdf').

Despite the aforementioned advantages, Bayesian methods—and Bayesian regression models (BRMs) in particular—are still not as common as their frequentist counterparts. In 2005, Woodward (2005) supposed one reason to be the lack of a "good user interface" which would allow applied researchers to fit BRMs as conveniently as other statistical methods for which a graphical user interface (GUI) already exists. In the meantime, several GUIs have emerged (see section Existing GUIs), but to our knowledge, until the first release of our R package **shinybrms** (Weber, 2022), there was no GUI which used Stan (Carpenter et al., 2017; Stan Development Team, 2022d) for inferring the posterior in BRMs. Stan has several advantages compared to other methods for inferring the posterior. In particular, it is highly flexible with respect to modeling choices and very efficient. Details will be given in section Algorithms for inferring the posterior.

Our **shinybrms** package is noncommercial and available at the Comprehensive R Archive Network (CRAN). While **shinybrms**'s frontend is a **shiny** (Chang et al., 2022) app, **shinybrms**'s backend completely relies on **brms** (Bürkner, 2017, 2018) which itself relies on Stan. Both of **brms**'s backends (i.e., interfaces to Stan), namely **rstan** (Stan Development Team, 2022b) and **cmdstanr** (Gabry and Češnovar, 2022), are supported by **shinybrms**. For the inspection of the Stan output, the **shinystan** (Gabry, 2022) app may be launched from within **shinybrms**.

To explain the particular advantages of Stan in detail, we have to take a closer look at different ways for inferring the posterior. This is the purpose of section Algorithms for inferring the posterior. In section Existing GUIs, we summarize existing GUIs for BRMs. That section is partly influenced by Ramírez-Hassan and Graciano-Londoño (2021). In section Features of shinybrms, we present the features of **shinybrms**. The usage of the **shinybrms** app is illustrated by the help of a real-world example in section Example. Finally, we discuss our work in section Discussion.

2 Algorithms for inferring the posterior

As mentioned above, in Bayesian statistics, uncertainty arising from the estimation of nuisance parameters is taken into account by integrating them out from the posterior. This is not the only integration occurring in posterior inference: Basically every quantity derived from the posterior is somehow connected to an integration over the posterior. However, it is the integration which also causes a lot of complications. While it is most desirable to perform posterior inference by exact

[2]Here, we are slightly abusing the notation by employing a single integral symbol for a possibly multiple integral.

calculation of the desired integrals (using analytic expressions), this approach is often infeasible and even if it is feasible, it has the downside of being not as flexible as other approaches since it needs to be tailored to the statistical model at hand. Numerical integration (e.g., by quadrature) may seem like a remedy, but is often only feasible up to a limited dimensionality of the parameter space. Depending on the algorithm, numerical integration may also introduce tuning quantities, hindering its "out-of-the-box" usage. Integration by simple Monte Carlo (MC) sampling may seem like an alternative, but this is only possible for distributions one may directly sample from (e.g., a Gaussian distribution).

Markov chain Monte Carlo

With the advent of Markov chain Monte Carlo (MCMC) methods, Bayesian inference has changed a lot (Woodward, 2005; Lunn et al., 2009). The first MCMC algorithm was the *Metropolis* algorithm (Metropolis et al., 1953) which starts from an initial point in parameter space and iteratively samples a *proposal*[3] from a *symmetric*[4] jumping distribution and accepts the proposal with a certain acceptance probability which depends on the ratio of the target (here, the posterior) density at the current position and at the proposal. The *Metropolis-Hastings (MH)* algorithm (Metropolis et al., 1953; Hastings, 1970) generalizes the Metropolis algorithm to asymmetric jumping distributions. *Gibbs sampling* (Geman and Geman, 1984; Gelfand and Smith, 1990) consists of alternately sampling from the full conditional posterior distributions and is a special MH algorithm in which the proposal is always accepted (Gelman et al., 2014). Combinations of the aforementioned algorithms are also widely used, e.g., MH-within-Gibbs. All MCMC algorithms (including those mentioned hereafter) require a careful examination of the convergence of the Markov chains. The MCMC diagnostics used for this purpose in **shinybrms** are outlined in section Tab "MCMC diagnostics".

Hamiltonian Monte Carlo (HMC) (initial work and major contributions by Duane et al., 1987; Neal, 1993; MacKay, 2003; Neal, 2011) is a special MCMC algorithm which is often more efficient than other MCMC algorithms, especially in case of a high-dimensional posterior distribution and correlated parameters (Hoffman and Gelman, 2014; Betancourt, 2018). The efficiency of HMC is due to the fact that it takes advantage of the gradient of the (log) posterior density (Stan Development Team, 2022a), making it a combination of stochastic and deterministic procedures (which explains why HMC is also known as *hybrid Monte Carlo*) (Gelman et al., 2014). HMC provides helpful diagnostics, such as divergent transitions which can (but must not necessarily) indicate areas of the posterior which are hard to explore by the HMC sampler (Betancourt, 2018; Gabry et al., 2019). Compared to Gibbs sampling, HMC also has the advantage that nonconjugate priors may be used easily. For the original HMC algorithm, three tuning quantities need to be specified by hand in advance: the *mass matrix* $\boldsymbol{M}$ (which is the covariance matrix of the auxiliary momentum vector), the number L of *leapfrog steps*, and the size ϵ of the leapfrog steps (Gelman et al., 2014; Stan Development Team, 2022a).

Because of the fixed choice of L, the original HMC algorithm is a *static* HMC algorithm (Betancourt, 2018). In contrast, the *no-U-turn sampler (NUTS)* (Hoffman and Gelman, 2014) is a *dynamic* HMC algorithm since it automatically chooses a (possibly) new value of L in each iteration of each Markov chain. Hoffman and Gelman (2014) also proposed a new dual averaging technique for determining ϵ automatically, too. Apart from these automations, the NUTS has the advantage that in terms of efficiency, it was shown to perform as well as—or even better than—a well-tuned static HMC algorithm (Hoffman and Gelman, 2014). A modified (Betancourt, 2018) NUTS is implemented in Stan. Stan's NUTS also includes an automatic adaptation of the mass matrix $\boldsymbol{M}$ during the warmup phase (Stan Development Team, 2022a). A complete presentation of Stan's NUTS is out of the scope of this article. A good starting point for a detailed description is Stan Development Team (2022a) as well as Betancourt (2018). Note that Stan also includes other algorithms for inferring or approximating the posterior. In this paper however, we only refer to Stan's NUTS when referring to Stan.

3 Existing GUIs

Table 1 summarizes existing GUIs for BRMs. Details are provided in Supplement section "Existing GUIs". Table 1 makes it clear that none of the existing GUIs relies *entirely* on Stan or the NUTS. JASP does use Stan for some analyses, but JASP's concept is quite different from **shinybrms**'s concept: While JASP offers a plenty of different statistical methods (including non-regression analyses), **shinybrms** is designed to be as concise as possible. While JASP's approach of using Stan for only some analyses certainly has a few advantages (especially in terms of runtime), **shinybrms**'s approach of completely relying on Stan (and **brms** in particular) has the advantage of a better maintainability: **shinybrms** only

[3]Here, the term "proposal" refers to a proposed parameter vector.

[4]Here, the term "symmetric" refers to the preservation of the distribution when reverting the jump, not to the symmetry in the shape of a distribution.

Table 1: Existing GUIs for BRMs. Algorithmic details for the GUIs may be found in Supplement section "Existing GUIs". "BB" stands for "Bayesian bootstrap". Here, the term "NUTS" includes Stan's NUTS. The column "Algorithm choice" specifies if given a model, the user may choose an algorithm (at least for some types of models). Notes: (i) The TEET package is noncommercial, but MATLAB is commercial; (ii) JASP uses (Stan's) NUTS for Bayesian meta-analyses and mixed BRMs; (iii) For linear regression models, BEsmarter offers a choice between MCMC and the Bayesian bootstrap.

		Algorithm (for inferring the posterior)							
		Non-MCMC				MCMC			
GUI name	Commercial	Analytic	Numerical	MC	BB	Non-HMC	Static HMC	NUTS	Algorithm choice
WinBUGS (Lunn et al., 2000)	no	no	no	no	no	yes	no	no	no
OpenBUGS (Spiegelhalter et al., 2014)	no	no	no	no	no	yes	yes	no	no
BugsXLA (Woodward, 2011)	no	no	no	no	no	yes	yes	no	no
IBM SPSS Amos (Arbuckle, 2020)	yes	no	no	no	no	yes	yes	no	yes
TEET (Qian, 2011)	no(i)	yes	yes	yes	no	yes	no	no	no
JASP (JASP Team, 2022)	no	yes	yes	yes	no	yes	no	yes(ii)	no
BRNPM (Karabatsos, 2015, 2017)	no	no	no	no	no	yes	no	no	no
Stata (StataCorp, 2019b)	yes	no	no	no	no	yes	no	no	yes
BayES (Emvalomatis, 2020)	no	no	no	no	no	yes	no	no	no
IBM SPSS (IBM Corp., 2020)	yes	yes	yes	yes	no	no	no	no	no
BEsmarter (BEsmarter Team, 2020a,b; Ramírez-Hassan and Graciano-Londoño, 2021)	no	no	no	no	yes	yes	no	no	yes(iii)

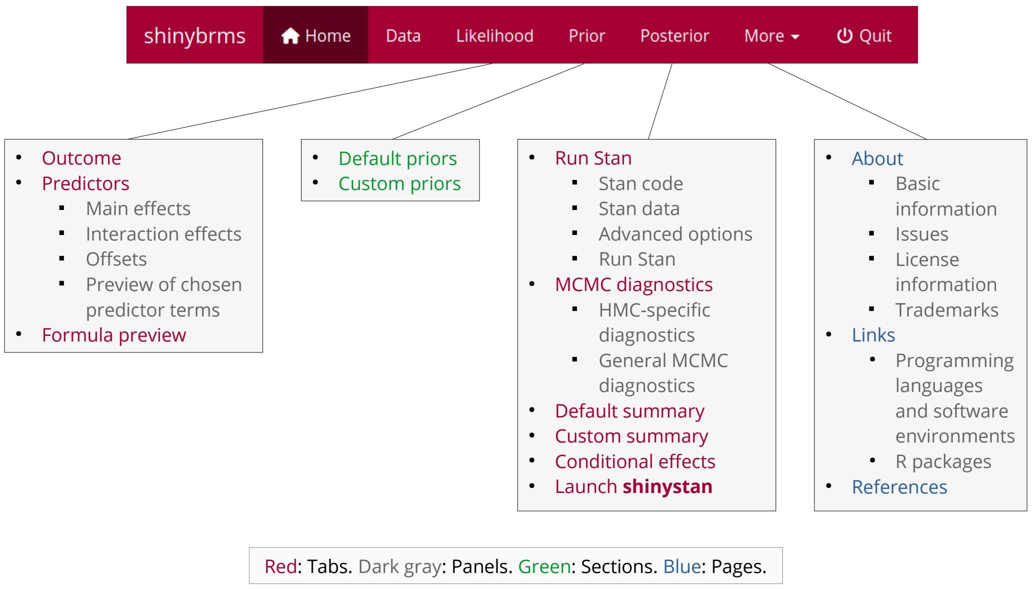


Figure 1: The navigation bar in **shinybrms**. In this figure, we have expanded the navigation bar itself by the structure of the three main pages ("Likelihood", "Prior", and "Posterior") as well as of the drop-down menu "More".

provides a lightweight GUI and only needs to perform few computations on its own. This is due to **brms** which is very flexible by allowing to fit a variety of regression models within a single R package. This division of work between **shinybrms**, **shiny**, **brms**, **rstan**, Stan, and **shinystan** reduces the amount of maintenance necessary for **shinybrms**, resulting in a faster integration of new features, a faster elimination of bugs, and a longer life cycle. Furthermore, it allows the authors of each component to focus on their strengths.

4 Features of shinybrms

The following general presentation of **shinybrms**'s features will be in written form, but with links to the corresponding screenshots from the example in section Example. In this article, not all aspects of the **shinybrms** app are shown in screenshots. For more screenshots, see the **shinybrms** website (Weber, 2022).

Note that the mathematical formulation of the models which may be fit with **shinybrms** has already been given elsewhere (Bürkner, 2017, 2018), so we will keep it short here.

Overview

The **shinybrms** app has three main pages which are accessible from a navigation bar at the top (Figure 1): "Likelihood", "Prior", and "Posterior". This structure follows Bayes' theorem, simplified to the proportionality of the posterior density to the product of prior density and likelihood:

$$p(\boldsymbol{\theta}|\boldsymbol{\mathcal{D}}) \propto p(\boldsymbol{\theta}) \cdot p(\dot{\boldsymbol{\mathcal{D}}}|\boldsymbol{\theta}, \ddot{\boldsymbol{\mathcal{D}}}) \tag{2}$$

where we have split up the data $\boldsymbol{\mathcal{D}}$ into $\boldsymbol{\mathcal{D}} = \begin{bmatrix} \dot{\boldsymbol{\mathcal{D}}} & \ddot{\boldsymbol{\mathcal{D}}} \end{bmatrix}$ because in BRMs, the distribution in the likelihood typically conditions on the predictor part $\ddot{\boldsymbol{\mathcal{D}}}$ of the data (see section Tab "Predictors" below). In the following sections, these three main pages will be described in detail.

There are also some auxiliary pages, the first two having direct links in the navigation bar, the last three being accessible from the drop-down menu "More" at the end of the navigation bar:

- The starting page "Home" gives a short overview of **shinybrms**'s objective and structure. Thus, it only contains informational text and no interactive elements.
- On page "Data", the user uploads his or her custom dataset which shall be used for the regression analysis. For testing purposes, page "Data" also offers example datasets. The chosen dataset (no matter if it was uploaded or chosen from the list of example datasets) is automatically shown in a preview consisting of the dataset's first six rows (there is an option to show the full dataset,

though). It is also possible to show the output of R's `str()` function applied to the chosen dataset, which gives some basic information about the dataset and its variables for users familiar with R.

- Page "About" contains basic information about **shinybrms** (e.g., version and corresponding date) as well as some legal information.
- Page "Links" gives links to software relevant for the **shinybrms** app.
- Page "References" contains the references for literature cited throughout the app.

Page "Likelihood"

Page "Likelihood" has three tabs: "Outcome", "Predictors", and "Formula preview". These tabs will now be described in turn.

Tab "Outcome"

On tab "Outcome" (Figure 2), the user specifies the outcome variable $\boldsymbol{y} = (y_1, \ldots, y_N)^\intercal \in \mathbb{R}^N$ (by choosing it from a drop-down list of the variables present in the dataset) as well as its distributional family, i.e., the basic form of the likelihood $p(\dot{\boldsymbol{\mathcal{D}}}|\boldsymbol{\theta}, \ddot{\boldsymbol{\mathcal{D}}})$, now with $\dot{\boldsymbol{\mathcal{D}}} = \boldsymbol{y}$. For the distributional family, there is a drop-down menu and a checkbox called "Show advanced distributional families". By default, this checkbox is unchecked which means that the drop-down menu offers three general distributional families of broad practical relevance: the Gaussian family (with the identity link function), the Bernoulli family with the logit link function, and the negative binomial family with the log link function. This is intended to be a limited selection: By reducing the choices as much as possible, we want to avoid overwhelming the user with a variety of special distributions. For example, the Poisson family is intentionally left out, in favor of the more general negative binomial distribution. However, by checking the "Show advanced distributional families" checkbox, the drop-down menu is extended so that a variety of other distributional families can be selected as well (see Supplement section "Advanced distributional families").

Tab "Predictors"

If desired, the user may specify predictors on tab "Predictors" (Figure 3). We use the term "predictor" for a column in the model matrix. In contrast, we use the term "predictor variable" for a column in the input dataset. Thus, a predictor may also denote an interaction and a predictor variable with K categories leads to $K - 1$ predictors (due to dummy coding).

The **shinybrms** app supports population-level effects as well as group-level effects[5]. The inclusion of group-level effects yields a multilevel model (also known as hierarchical or mixed-effects model). Here, we denote the vector of population-level effects by $\boldsymbol{\beta}$ and the corresponding model matrix by $\boldsymbol{X}$. Likewise, we denote the vector of group-level effects by $\boldsymbol{u}$ and the corresponding model matrix by $\boldsymbol{Z}$. The hyperparameters for the group-level effects (i.e., their standard deviations and correlations) will be collected in a vector $\boldsymbol{\tau}$. If the model does not contain group-level effects, we define here (for the mathematical description, not for the software) $\boldsymbol{u} = 0$, $\boldsymbol{Z} = (0, \ldots, 0)^\intercal \in \mathbb{R}^N$ (for example; the exact values in $\boldsymbol{Z}$ do not matter if $\boldsymbol{u} = 0$), and $\boldsymbol{\tau} = 0$ (for example). With $\boldsymbol{X}$ and $\boldsymbol{Z}$, we now have $\ddot{\boldsymbol{\mathcal{D}}} = \begin{bmatrix}\boldsymbol{X} & \boldsymbol{Z}\end{bmatrix}$. We denote the vector of linear predictors by $\boldsymbol{\eta} = \boldsymbol{X\beta} + \boldsymbol{Zu} \in \mathbb{R}^N$ (written out as $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_N)^\intercal$).

Note that here as well as in **shinybrms**, the term "interaction" is also used for interactions involving predictor variables with group-level main effects (yielding group-level interaction effects). This broad definition of "interaction" simplifies the GUI and emphasizes the key concept of interactions, namely that an effect depends on another predictor (or on other predictors).

To avoid common mistakes, **shinybrms** imposes some restrictions: Firstly, an overall (population-level) intercept is always included. Secondly, including an interaction causes all corresponding lower-order interactions to be automatically included, too. The latter restriction also implies that interactions may only involve predictor variables for which main effects have already been added.

[5]Population-level effects are also known as fixed effects (Bürkner, 2017, 2018). Group-level effects are also known as random or partially pooled effects (Bürkner, 2017, 2018; Goodrich et al., 2022). The terms "fixed" and "random" effects are not really appropriate in a Bayesian context: In a Bayesian model, all parameters have a prior distribution and may therefore be considered as random (Marchenko and Balov, 2015).

Tab "Formula preview"

The tab "Formula preview" simply combines the chosen outcome and the chosen predictors into **brms**'s formula syntax. This is mainly intended for checking the correct specification of the model formula (for users familiar with the syntax). However, it also provides a concise and standardized way to communicate this central part of the model because together with the distributional family, this model formula determines the likelihood $p(\dot{\boldsymbol{\mathcal{D}}}|\boldsymbol{\theta}, \ddot{\boldsymbol{\mathcal{D}}}) = p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{Z})$: In case of the Gaussian family (with the identity link function), we have

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi} \cdot \sigma} \exp\left(-\frac{1}{2} \cdot \left(\frac{y_i - \mu_i}{\sigma}\right)^2\right) \tag{3}$$

with $\mu_i = \eta_i$ (see $\boldsymbol{\eta}$ from section Tab "Predictors"), $\sigma \in (0, \infty)$, and $\boldsymbol{\theta} = (\boldsymbol{\beta}^\intercal, \boldsymbol{u}^\intercal, \boldsymbol{\tau}^\intercal, \sigma)^\intercal$. Note that the dependence on $\boldsymbol{X}$ and $\boldsymbol{Z}$ as well as on most elements of $\boldsymbol{\theta}$ is an indirect dependence *via* $\mu_1, \ldots, \mu_N$. In case of the Bernoulli family with the logit link function, we have $\boldsymbol{y} \in \{0,1\}^N$ and

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{N} \mu_i^{y_i} (1 - \mu_i)^{(1-y_i)} \tag{4}$$

with $\mu_i = \frac{1}{1+\exp(-\eta_i)}$ and $\boldsymbol{\theta} = (\boldsymbol{\beta}^\intercal, \boldsymbol{u}^\intercal, \boldsymbol{\tau}^\intercal)^\intercal$. In case of the negative binomial family with the log link function, we have $\boldsymbol{y} \in (\{0\} \cup \mathbb{N})^N$ and

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{X}, \boldsymbol{Z}) = \prod_{i=1}^{N} \binom{y_i + \zeta - 1}{y_i} \left(\frac{\mu_i}{\mu_i + \zeta}\right)^{y_i} \left(\frac{\zeta}{\mu_i + \zeta}\right)^{\zeta} \tag{5}$$

with $\mu_i = \exp(\eta_i)$, $\zeta \in (0, \infty)$, and $\boldsymbol{\theta} = (\boldsymbol{\beta}^\intercal, \boldsymbol{u}^\intercal, \boldsymbol{\tau}^\intercal, \zeta)^\intercal$. The mathematical details of the "advanced" distributional families (see section Tab "Outcome" above) may be found in the **brms** vignette "Parameterization of Response Distributions in brms".

Page "Prior"

At the top of page "Prior", the user obtains a preview of the default priors taken from **brms** (Figure 4). At the bottom, the user may specify custom priors (Figure 5). Both, the default and the custom priors, refer to the parameters of the currently specified likelihood. Custom priors may be specified as follows:

- *via* a Stan function,
- *via* one of the special **brms** (pseudo-)functions designed for this purpose, e.g., for the Lewandowski-Kurowicka-Joe (LKJ) prior (Lewandowski et al., 2009),
- *via* an empty input field to specify a flat prior over the whole support of the corresponding parameter(s).

The first two possibilities always lead to a proper prior. The flat prior is only proper if the support is bounded on both sides. Otherwise (which is the more common case), the flat prior is improper.

The user's selections on page "Prior" ultimately lead to the specification of $p(\boldsymbol{\theta})$. Together with the likelihood, this completes the model specification. Note that for multilevel models, $p(\boldsymbol{\theta})$ here[6] includes the distributions of the group-level effects $\boldsymbol{u}$ as well, even though they do not need to be specified on page "Prior".

Page "Posterior"

Page "Posterior" has six tabs: "Run Stan", "MCMC diagnostics", "Default summary", "Custom summary", "Conditional effects", and "Launch **shinystan**". These will now be described in turn. The output shown on these tabs may also be downloaded (with the file format depending on the specific type of output).

Tab "Run Stan"

At the top of tab "Run Stan", the user may inspect and download the Stan code and the so-called *Stan data*. The Stan data basically consists of the pre-processed part of the chosen dataset which is

[6]In multilevel models, drawing the line between prior and likelihood can be done in multiple ways, so our mathematical formulation is just one of several possibilities.

needed for the Stan model, extended by some internal objects. Apart from checking or documentation purposes, the Stan code and the Stan data are needed if the user wants to customize the Stan code and then run Stan outside of **shinybrms**.

Further down on tab "Run Stan", the user may set advanced options for the Stan run (Figure 6). These options have sensible defaults, but sometimes they need to be changed. Probably the most important option is the seed for the pseudorandom number generator.

The final panel on tab "Run Stan" is the central one (Figure 7): By a click on the "Run Stan" button, Stan translates the Stan code written by **brms** to C++ code, compiles this C++ code, and then starts sampling. By default, Stan writes its sampling progress to an HTML file which is automatically opened up by **shinybrms**. The user then only needs to refresh this HTML file to see the current sampling progress. An example for Stan's runtime will be given in section Example.

When Stan has finished sampling, the panel "Run Stan" automatically refreshes, in particular to show the result from an overall check of the MCMC diagnostics (see section Tab "MCMC diagnostics" for details). The user also has the possibility to download different output objects which can be analyzed outside of **shinybrms** or—in case of the fitted model object of class `"brmsfit"`—uploaded in a new **shinybrms** session (to avoid re-running Stan).

Tab "MCMC diagnostics"

On tab "MCMC diagnostics" (Figure 8), the user obtains detailed information concerning the following MCMC diagnostics:

- HMC-specific diagnostics:
  - the number of iterations ending with a divergence,
  - the number of iterations hitting the maximum tree depth,
  - the Bayesian fraction of missing information for the energy transitions (E-BFMI);
- some general MCMC diagnostics (which are computed for each parameter as well as for the accumulated log posterior density):
  - the modified potential scale reduction factor $\widehat{R}$ proposed by Vehtari et al. (2021) (here simply called *the* $\widehat{R}$ instead of *the modified* $\widehat{R}$)[7],
  - the effective sample size (ESS) in the bulk of the corresponding marginal posterior (Vehtari et al., 2021),
  - the ESS in the tails of the corresponding marginal posterior (Vehtari et al., 2021).

As a full description of these MCMC diagnostics is out of the scope of this article, we refer the interested reader to Stan Development Team (2022c), Betancourt (2018), and Vehtari et al. (2021). The most important basic guidelines for deciding whether these MCMC diagnostics are worrying are explained in the **shinybrms** GUI and also checked automatically by **shinybrms**. For the general MCMC diagnostics, it is also possible to show a detailed table with the diagnostics for each parameter (as well as for the accumulated log posterior density).

Tab "Default summary"

Tab "Default summary" (Figure 9) shows **brms**'s standard *robust* summary of the posterior inference, e.g., the medians and the central 95 % intervals of the marginal posteriors (the 95 % CrIs) of the most important parameters. This tab is only intended for a quick inspection. A much more comprehensive analysis of the Stan output is offered by the **shinystan** app (see section Tab "Launch shinystan").

Tab "Custom summary"

On tab "Custom summary" (Figure 10), the user may calculate posterior summary quantities for a custom mathematical (or logical) expression involving at least one parameter. Such an expression may be, e.g., a sum of two parameters (as shown in Figure 10) or the event that a parameter exceeds a certain threshold.

[7] The term "potential scale reduction factor" is not always appropriate (Vehtari et al., 2021, section 2), but because of its widespread use, we employ it here nonetheless.

Tab "Conditional effects"

On tab "Conditional effects" (Figure 11), **shinybrms** offers *conditional-effects plots* (created by `brms::conditional_effects()`). A conditional-effects plot shows the estimated effect of a predictor variable on the outcome. An interaction effect involving at most two predictor variables may also be visualized by showing the estimated effect of the first predictor variable separately for appropriate values of the second predictor variable.

As described in more detail in the **shinybrms** GUI, a conditional-effects plot *conditions* on specific values of those predictor variables which are not involved in the plot. Likewise, group-level effects which are not involved in the plot are (usually) set to zero.

Tab "Launch shinystan"

The **shinystan** app (Gabry, 2022) offers an interactive inspection of Stan (and other MCMC-generated) results, in particular with respect to:

- MCMC diagnostics (including several additional diagnostics not covered by **shinybrms**'s tab "MCMC diagnostics"),
- PPCs,
- summary quantities of univariate marginal posteriors,
- plots of univariate, bivariate, and trivariate marginal posteriors.

Before launching the **shinystan** app from within the **shinybrms** app by clicking the corresponding button on tab "Launch **shinystan**", the user may set a seed to ensure the reproducibility of the PPCs.

At this point, the **shinybrms** workflow ends and passes over to the **shinystan** workflow. We will illustrate the **shinystan** workflow in section Example.

5 Example

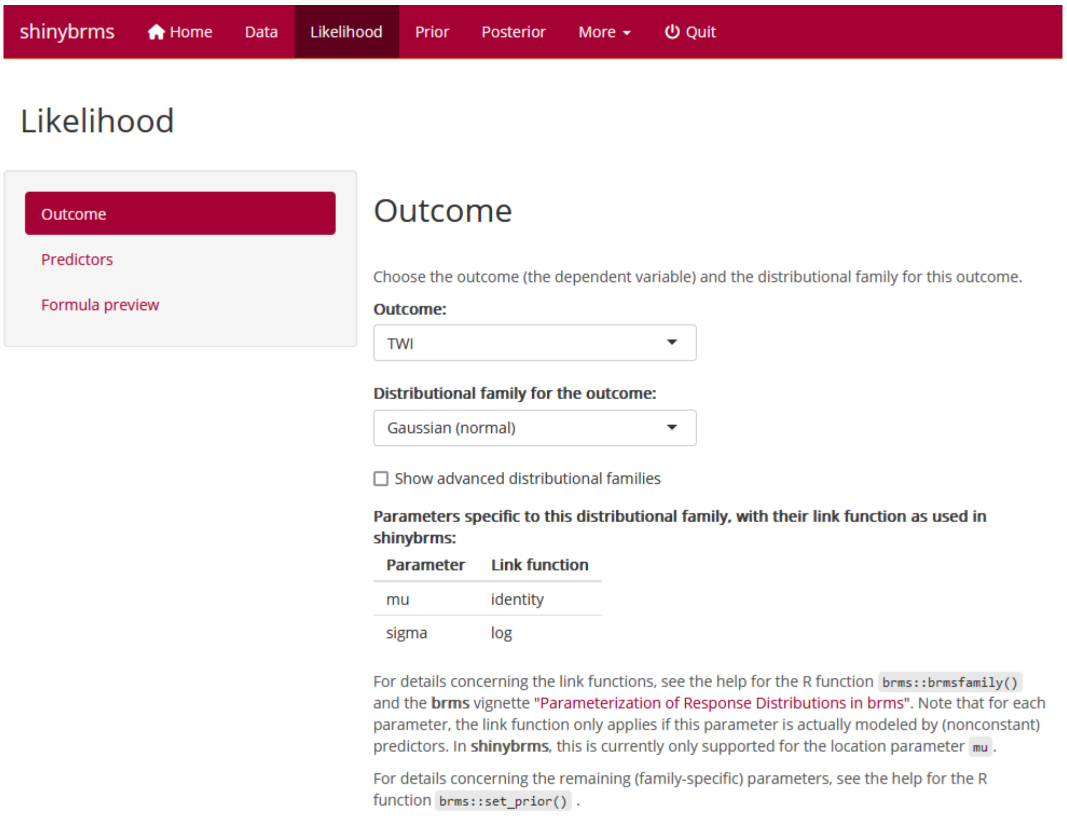
We illustrate **shinybrms**'s features following the workflow implied by Figure 1 and using a real-world dermatological dataset from Van Welzen et al. (2021). This dataset is available in the Supplement (file 'CAP.csv'). In Supplement section "Frequentist analysis of the example", we compare the Bayesian analysis presented here with a frequentist one, referring to the list of advantages of Bayesian statistics from section Introduction.

Van Welzen et al. (2021) conducted a prospective pilot study investigating the efficacy and safety of a novel cold atmospheric plasma (CAP) wound dressing for the healing of split-skin graft donor sites. The only outcome we focus on here is the tissue water index (TWI), measured by a hyperspectral imaging camera and having values between 0 and 100, with lower TWI values being associated with an improved wound healing. Briefly, the study design was as follows: For each of $P = 10$ patients, the TWI was measured under $T = 3$ different treatment conditions (standard-treated wound, CAP-treated wound, and healthy skin). Each treatment condition was investigated in its own skin area with $R = 3$ measurements across that area. This procedure of measuring was repeated on each of $D = 4$ days (day 1, 3, 5, and 7, with day 1 being the day of the split-skin graft donation where the TWI was measured *after* the split-skin graft donation but *before* the first wound dressing). Thus, the dataset consists of $N = P \cdot T \cdot R \cdot D = 360$ observations (rows). The dataset's columns are:

- `patID` (for "patient ID"; coded as `"pat1"`, ..., `"pat10"`),
- `age` (in years),
- `anticoagulation` (indicating whether the patient received an anticoagulation therapy before and during the study; coded as `"no"` and `"yes"`),
- `diabetes` (indicating whether the patient is diabetic; coded as `"no"` and `"yes"`),
- `day` (coded as `"d1"`, `"d3"`, `"d5"`, and `"d7"`),
- `trt` (coded as `"0_standard"`, `"CAP"`, and `"healthy"` to make `"0_standard"` the reference level),
- `TWI` (integers in the interval $[0, 100]$).

The primary research question is whether the CAP treatment leads to a decreased[8] TWI compared to the standard treatment (polyhexanide wound gel with fatty gauze). The healthy skin area serves as an *experimental* control (albeit not as a control *treatment* since this is the role of the standard-treated wound area) and is not part of the primary research question.

[8]For this demonstration here, we won't discuss whether this decrease is clinically relevant.



Likelihood

Outcome

Choose the outcome (the dependent variable) and the distributional family for this outcome.

Outcome: TWI

Distributional family for the outcome: Gaussian (normal)

☐ Show advanced distributional families

Parameters specific to this distributional family, with their link function as used in shinybrms:

Parameter	Link function
mu	identity
sigma	log

For details concerning the link functions, see the help for the R function `brms::brmsfamily()` and the **brms** vignette "Parameterization of Response Distributions in brms". Note that for each parameter, the link function only applies if this parameter is actually modeled by (nonconstant) predictors. In **shinybrms**, this is currently only supported for the location parameter `mu`.

For details concerning the remaining (family-specific) parameters, see the help for the R function `brms::set_prior()`.

Figure 2: Tab "Outcome" on page "Likelihood". In the example presented here, we select `TWI` as the outcome variable and the Gaussian family as the distributional family for this outcome. Tab "Outcome" and tab "Predictors" (Figure 3) are the two main components of page "Likelihood".

shinybrms

After launching the **shinybrms** app in R *via*

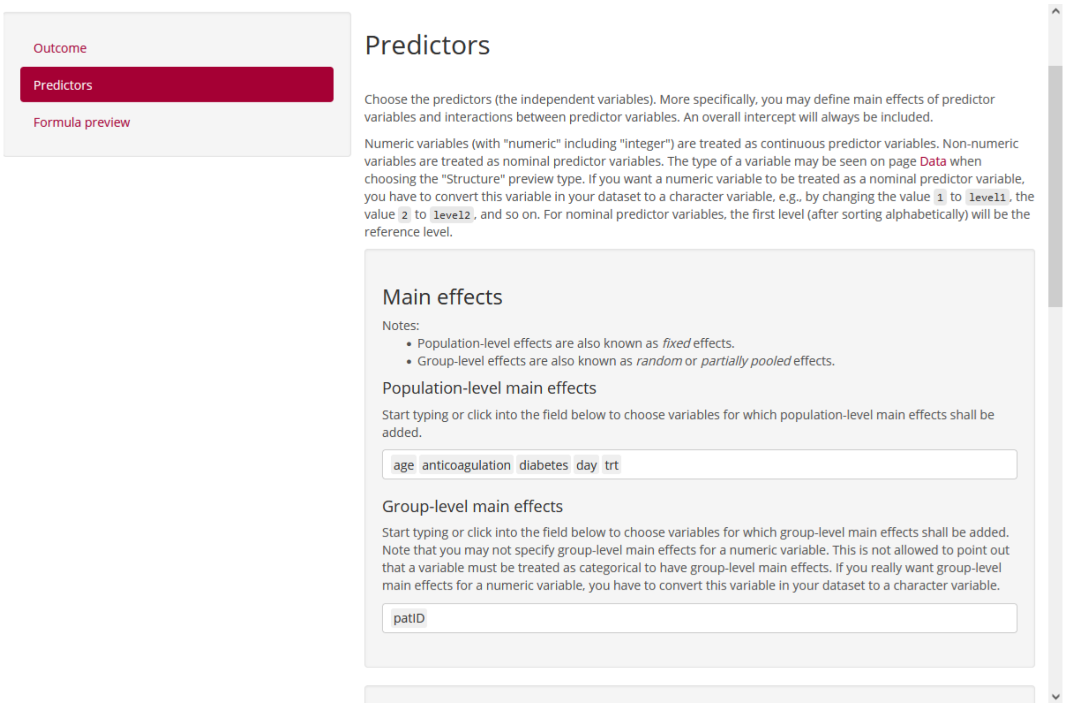
```
> library("shinybrms")
> launch_shinybrms(launch.browser = TRUE)
```

(with `launch.browser` set to `TRUE` to ensure that the app is opened up in the default web browser), we switch to page "Data" where we upload the dataset (not shown here).

Next, we head over to page "Likelihood". On tab "Outcome" (Figure 2), we choose the outcome variable `TWI` and the Gaussian family as the distributional family for this outcome. Clearly, the TWI values cannot follow an unmodified Gaussian distribution since they are bounded by 0 and 100, with the minimum of the observed TWI values being indeed as low as 9 (the maximum being 67). Thus, a truncated Gaussian distribution might be more appropriate here. We will come back to this later in section Discussion.

On tab "Predictors" (Figure 3), we choose `age`, `anticoagulation`, `diabetes`, `day`, and `trt` to have population-level main effects and `patID` to have group-level main effects ("random intercepts"). Further down on tab "Predictors", we add an interaction between `day` and `trt` (not shown here). This interaction is included because the TWI is supposed to show a stronger time-dependence in the two wound areas than in the healthy skin area. Additionally, the TWI difference (in means) between the standard and the CAP treatment might change over time.

Now the likelihood is set up, so we can proceed with the prior. The default priors (Figure 4) are reasonable, but suppose we wanted a weakly informative Student-t prior with 3 degrees of freedom, a location parameter of 0, and a scale parameter of 30 for all regression coefficients. To add this custom prior, we choose parameter class `b` from the corresponding drop-down list shown in Figure 5, enter `student_t(3, 0, 30)` into the input field entitled "Prior distribution", and click the "Add prior" button. After doing so, our Student-t prior is added to the preview table (Figure 5, right-hand side). For all remaining parameters for which we do not specify a custom prior, the corresponding default prior will be used.


Outcome
Predictors
Formula preview

Predictors

Choose the predictors (the independent variables). More specifically, you may define main effects of predictor variables and interactions between predictor variables. An overall intercept will always be included.

Numeric variables (with "numeric" including "integer") are treated as continuous predictor variables. Non-numeric variables are treated as nominal predictor variables. The type of a variable may be seen on page Data when choosing the "Structure" preview type. If you want a numeric variable to be treated as a nominal predictor variable, you have to convert this variable in your dataset to a character variable, e.g., by changing the value `1` to `level1`, the value `2` to `level2`, and so on. For nominal predictor variables, the first level (after sorting alphabetically) will be the reference level.

Main effects

Notes:
- Population-level effects are also known as *fixed* effects.
- Group-level effects are also known as *random* or *partially pooled* effects.

Population-level main effects

Start typing or click into the field below to choose variables for which population-level main effects shall be added.

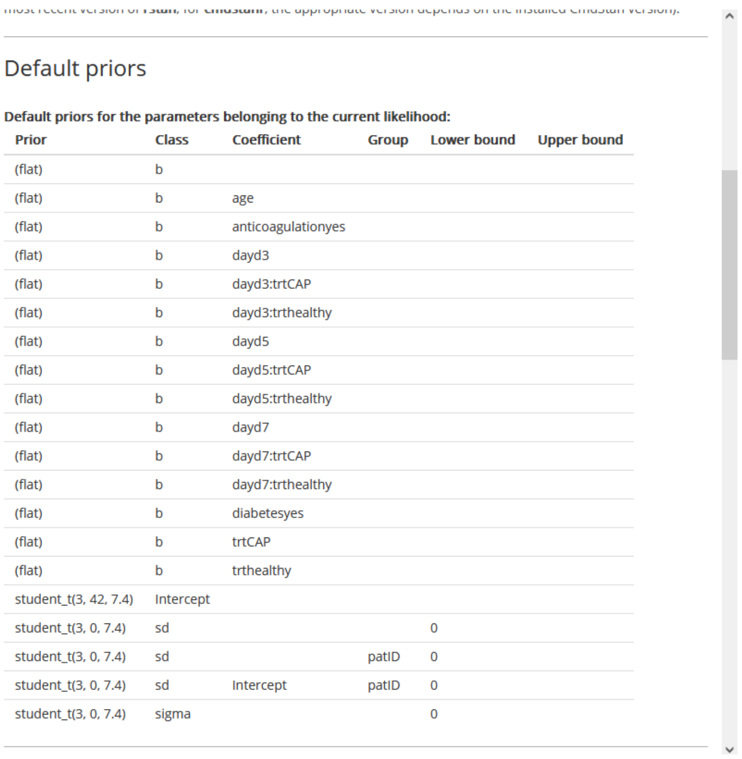
age anticoagulation diabetes day trt

Group-level main effects

Start typing or click into the field below to choose variables for which group-level main effects shall be added. Note that you may not specify group-level main effects for a numeric variable. This is not allowed to point out that a variable must be treated as categorical to have group-level main effects. If you really want group-level main effects for a numeric variable, you have to convert this variable in your dataset to a character variable.

patID

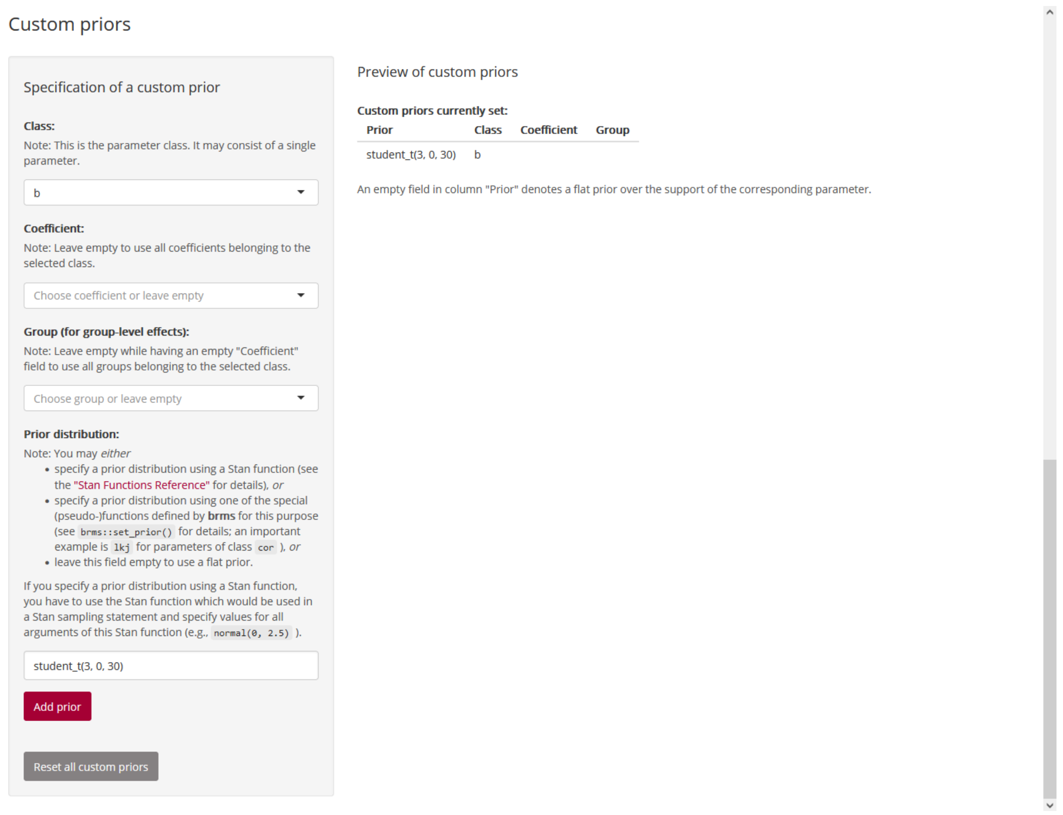
Figure 3: Tab "Predictors" on page "Likelihood". Here, the main effects of the predictors need to be defined first (in the example presented here: variables age, `anticoagulation`, `diabetes`, `day`, `trt`, and `patID`). Then, further down on this tab (not visible here), interactions can be specified (in the example presented here: an interaction between variables day and `trt`). In principle, offsets may also be specified further down on this tab (not visible here), but our example does not feature offsets. For the main effects, the user may choose between population-level and group-level effects. For interaction effects, this choice will be performed automatically based on the involved main effects.


Default priors

Default priors for the parameters belonging to the current likelihood:

Prior	Class	Coefficient	Group	Lower bound	Upper bound
(flat)	b				
(flat)	b	age			
(flat)	b	anticoagulationyes			
(flat)	b	dayd3			
(flat)	b	dayd3:trtCAP			
(flat)	b	dayd3:trthealthy			
(flat)	b	dayd5			
(flat)	b	dayd5:trtCAP			
(flat)	b	dayd5:trthealthy			
(flat)	b	dayd7			
(flat)	b	dayd7:trtCAP			
(flat)	b	dayd7:trthealthy			
(flat)	b	diabetesyes			
(flat)	b	trtCAP			
(flat)	b	trthealthy			
student_t(3, 42, 7.4)	Intercept				
student_t(3, 0, 7.4)	sd			0	
student_t(3, 0, 7.4)	sd		patID	0	
student_t(3, 0, 7.4)	sd	Intercept	patID	0	
student_t(3, 0, 7.4)	sigma			0	

Figure 4: Section "Default priors" on page "Prior". The default priors are taken from **brms** and depend on the currently specified likelihood. They can be overridden by custom priors (Figure 5).


Custom priors

Specification of a custom prior

Class:
Note: This is the parameter class. It may consist of a single parameter.

b

Coefficient:
Note: Leave empty to use all coefficients belonging to the selected class.

Choose coefficient or leave empty

Group (for group-level effects):
Note: Leave empty while having an empty "Coefficient" field to use all groups belonging to the selected class.

Choose group or leave empty

Prior distribution:
Note: You may *either*

- specify a prior distribution using a Stan function (see the "Stan Functions Reference" for details), *or*
- specify a prior distribution using one of the special (pseudo-)functions defined by **brms** for this purpose (see `brms::set_prior()` for details; an important example is `lkj` for parameters of class `cor`), *or*
- leave this field empty to use a flat prior.

If you specify a prior distribution using a Stan function, you have to use the Stan function which would be used in a Stan sampling statement and specify values for all arguments of this Stan function (e.g., `normal(0, 2.5)`).

student_t(3, 0, 30)

Add prior

Reset all custom priors

Preview of custom priors

Custom priors currently set:

Prior	Class	Coefficient	Group
student_t(3, 0, 30)	b		

An empty field in column "Prior" denotes a flat prior over the support of the corresponding parameter.

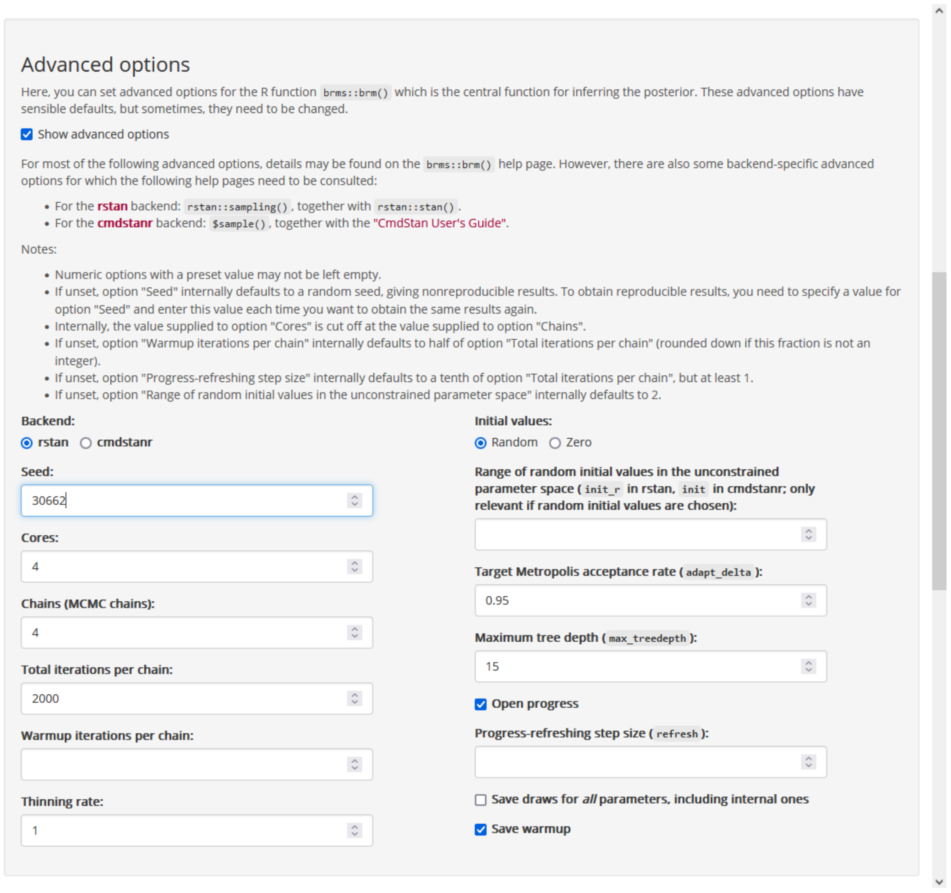
Figure 5: Section "Custom priors" on page "Prior". Here, we specify a Student-t prior with 3 degrees of freedom, a location parameter of 0, and a scale parameter of 30 for all regression coefficients (parameter class b). This overrides the default flat prior for these parameters (Figure 4).

Now the model is fully set up, so we can start inferring the posterior. To do this, we switch to page "Posterior" where we scroll down to the advanced options on tab "Run Stan" (Figure 6). There, we set a seed for reproducibility. Afterwards, we scroll further down to panel "Run Stan" where we click the button for starting the Stan run (Figure 7). For this example, the Stan run as a whole (including the compilation of the C++ code) takes about 50 seconds on a standard desktop machine.

After Stan has finished sampling, we receive a pop-up notification (not shown here) whether all MCMC diagnostics have passed their checks. Here, this is the case as we may also see on tab "MCMC diagnostics" (Figure 8).

Since **shinybrms** reports all MCMC diagnostics as being OK, we may start interpreting the posterior. On tab "Default summary" (Figure 9), it is mainly the summary of the population-level effects which is of interest here: With each additional year of age, the TWI is estimated to increase by ca. 0.10 with a 95 % CrI of ca. $(-0.28, 0.48)$. An `anticoagulation` therapy is estimated to increase the TWI by ca. -1.12 with a 95 % CrI of ca. $(-7.53, 5.69)$. A `diabetes` disease is estimated to increase the TWI by ca. -0.56 with a 95 % CrI of ca. $(-6.51, 5.66)$. As may be seen from these three CrIs, the statistical uncertainty is quite big which is probably due to the small $P = 10$.

Since we included an interaction between `day` and `trt`, the coefficients for these two variables are most conveniently interpreted by the help of a custom summary (Figure 10) and a conditional-effects plot (Figure 11). On tab "Custom summary" (Figure 10), we may calculate the estimated TWI difference (in means) between the CAP and the standard treatment separately for each day by entering the corresponding sum expressions (and the expression `` `b_trtCAP` `` for day 1) in turn. The resulting table is included in Figure 10: On day 1 (where the two wound areas had not been treated yet), the standard treatment and the CAP treatment lead to a quite similar TWI (the posterior median of their TWI difference being ca. -0.74 with a 95 % CrI of ca. $(-3.69, 2.13)$). In contrast, on days 3, 5, and 7, the CAP treatment clearly leads to a *lower* TWI than the standard treatment (posterior medians of ca. -10.45, -7.66, and -7.10, respectively, and 95 % CrIs of ca. $(-13.43, -7.36)$, $(-10.63, -4.84)$, and $(-10.11, -4.03)$, respectively). This answers the primary research question: The CAP treatment indeed leads to a decreased TWI and therefore an improved wound healing compared to the standard treatment. This is also well illustrated by the conditional-effects plot (Figure 11). The conditional-effects plot also confirms that the TWI in the healthy skin area does not change as heavily over time as in the two wound areas.



Advanced options

Here, you can set advanced options for the R function `brms::brm()` which is the central function for inferring the posterior. These advanced options have sensible defaults, but sometimes, they need to be changed.

[✓] Show advanced options

For most of the following advanced options, details may be found on the `brms::brm()` help page. However, there are also some backend-specific advanced options for which the following help pages need to be consulted:

- For the **rstan** backend: `rstan::sampling()`, together with `rstan::stan()`.
- For the **cmdstanr** backend: `$sample()`, together with the "CmdStan User's Guide".

Notes:

- Numeric options with a preset value may not be left empty.
- If unset, option "Seed" internally defaults to a random seed, giving nonreproducible results. To obtain reproducible results, you need to specify a value for option "Seed" and enter this value each time you want to obtain the same results again.
- Internally, the value supplied to option "Cores" is cut off at the value supplied to option "Chains".
- If unset, option "Warmup iterations per chain" internally defaults to half of option "Total iterations per chain" (rounded down if this fraction is not an integer).
- If unset, option "Progress-refreshing step size" internally defaults to a tenth of option "Total iterations per chain", but at least 1.
- If unset, option "Range of random initial values in the unconstrained parameter space" internally defaults to 2.

Backend: [✓ rstan] cmdstanr

Seed: 30662

Cores: 4

Chains (MCMC chains): 4

Total iterations per chain: 2000

Warmup iterations per chain:

Thinning rate: 1

Initial values: [✓ Random] Zero

Range of random initial values in the unconstrained parameter space (`init_r` in rstan, `init` in cmdstanr; only relevant if random initial values are chosen):

Target Metropolis acceptance rate (`adapt_delta`): 0.95

Maximum tree depth (`max_treedepth`): 15

[✓] **Open progress**

Progress-refreshing step size (`refresh`):

[ ] **Save draws for *all* parameters, including internal ones**

[✓] **Save warmup**

Figure 6: Panel "Advanced options" on tab "Run Stan" of page "Posterior". The defaults for these advanced options should be fine for most practical situations. In the example presented here, we only set a specific seed so that results are reproducible.

Finally, we switch to tab "Launch **shinystan**", enter a seed for the reproducibility of the PPCs (here, `63438`), and click on the button for launching **shinystan**.

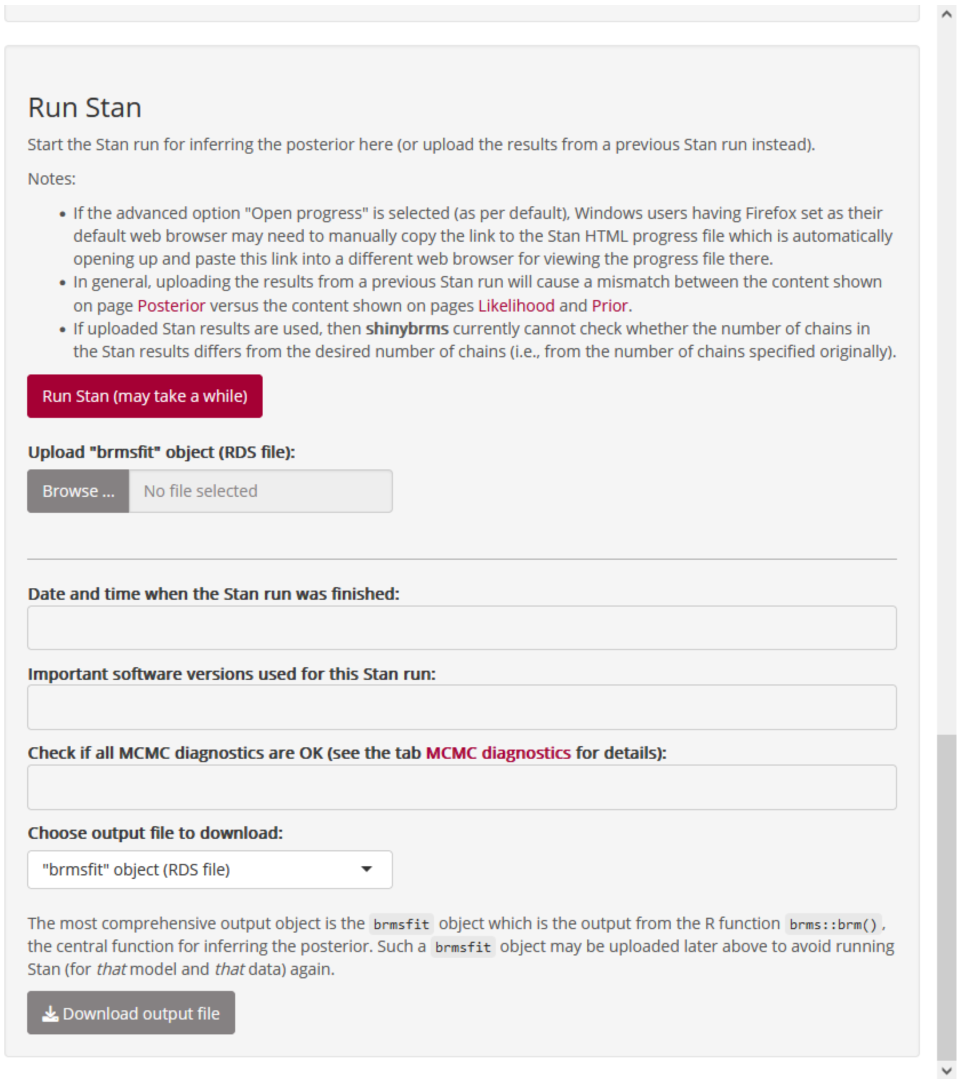
shinystan

Within **shinystan**, we may inspect some PPC plots, e.g., a kernel density estimate for the observed TWI values, overlaid by kernel density estimates for replicated TWI values (Figure 12). This overlaid density plot suggests that the model is appropriate, being able to generate outcome values similar to the observed ones after having estimated the unknown parameters by the help of the observed dataset (as well as the prior). Nevertheless, the model may still be improved, as illustrated by the PPC plots shown in Figure 13 (lower two histograms): The minimum of the observed TWI values is systematically smaller than the replicated minimums, the opposite holding—even if not that extremely—for the maximum. However, we consider the current model to be appropriate for the primary research question.

With respect to the parameter estimates, **shinystan** offers, e.g., a visualization of the posterior medians, together with 50 % and 95 % CrIs (Figure 14). The **shinystan** app also offers kernel density estimates for the univariate marginal posteriors (not shown).

6 Discussion

We have presented our **shiny** app called **shinybrms**, distributed as an R package. With the **shinybrms** GUI, we hope to make Bayesian regression modeling more accessible for people without any knowledge of R's syntax. Currently, the user still needs to execute some R code for setting up **shinybrms**'s backend and for launching the **shinybrms** app, even if he or she is using a GUI for installing R packages. We tried to make this as easy as possible by providing step-by-step instructions in the 'README' file of the **shinybrms** package. More importantly however, the **shinybrms** app may be hosted on a



Run Stan

Start the Stan run for inferring the posterior here (or upload the results from a previous Stan run instead).

Notes:

- If the advanced option "Open progress" is selected (as per default), Windows users having Firefox set as their default web browser may need to manually copy the link to the Stan HTML progress file which is automatically opening up and paste this link into a different web browser for viewing the progress file there.
- In general, uploading the results from a previous Stan run will cause a mismatch between the content shown on page Posterior versus the content shown on pages Likelihood and Prior.
- If uploaded Stan results are used, then **shinybrms** currently cannot check whether the number of chains in the Stan results differs from the desired number of chains (i.e., from the number of chains specified originally).

Run Stan (may take a while)

Upload "brmsfit" object (RDS file):

Browse ... No file selected

Date and time when the Stan run was finished:

Important software versions used for this Stan run:

Check if all MCMC diagnostics are OK (see the tab MCMC diagnostics for details):

Choose output file to download:

"brmsfit" object (RDS file)

The most comprehensive output object is the `brmsfit` object which is the output from the R function `brms::brm()`, the central function for inferring the posterior. Such a `brmsfit` object may be uploaded later above to avoid running Stan (for *that* model and *that* data) again.

Download output file

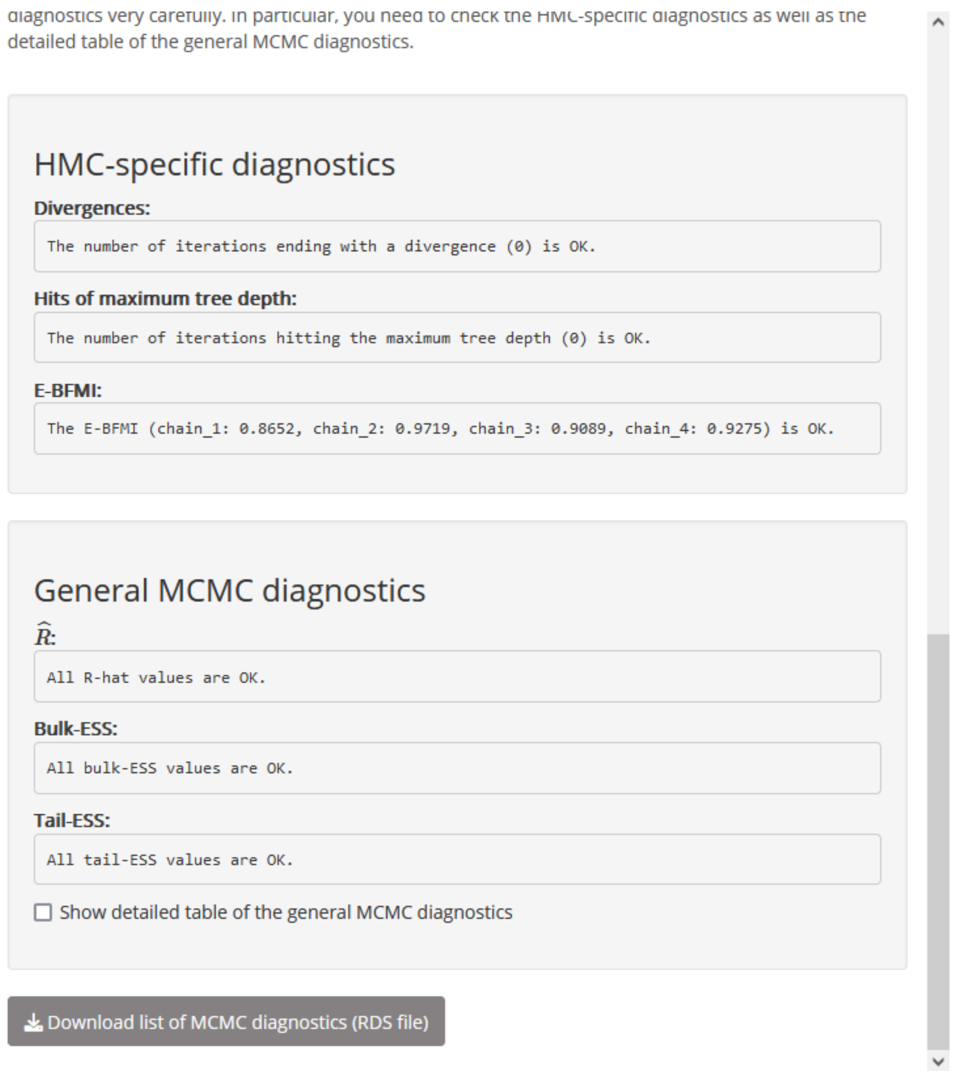
Figure 7: Panel "Run Stan" on tab "Run Stan" of page "Posterior". This is the central UI element: By clicking the red button, the Stan run is started, which first involves several preparation steps (including the compilation of the C++ code) and then the MCMC sampling itself.

server and accessed through a web browser, just like any other **shiny** app. In that case, the user does not need to install the **shinybrms** package or any other additional software. With the server-sided hosting, the **shinybrms** app may even be accessed from a mobile device where it is usually impossible to install any software designed for personal computers. Of course, setting up the server-sided hosting is a lot more complex than following the instructions from our 'README' file for running **shinybrms** on a local computer, but the idea is that IT departments of bigger institutions could establish the server-sided hosting (potentially adding an access control on top) and then members of that institution could access the **shinybrms** app through their web browsers.

Note that JASP offers an alternative host-client service by relying on rollApp (rollApp, Inc., 2020; rollApp, Inc. and JASP Team, 2020).

Apart from application in practice, **shinybrms** may also be valuable for teaching Bayesian regression models, e.g., to undergraduate students.

Of course, **shinybrms** may still be extended. As may be seen from our real-world example in section Example, truncated outcome families would be a useful feature. Apart from this, our future plans also include further outcome families supported by **brms** (e.g., ordinal and time-to-event regression), model selection features (e.g., using the package **projpred** by Piironen et al., 2022), and support for special **brms** features such as smoothed effects and known measurement error in the outcome variable (needed for meta-analyses). When implementing new features, the challenge will be to keep the GUI as simple as possible: In our opinion, a GUI such as **shinybrms** should support the user by automizing steps wherever this is appropriate and thus focus the attention to steps which may not be automized (in particular those related to the original research question).


diagnostics very carefully. In particular, you need to check the HMC-specific diagnostics as well as the detailed table of the general MCMC diagnostics.

HMC-specific diagnostics

Divergences:

```
The number of iterations ending with a divergence (0) is OK.
```

Hits of maximum tree depth:

```
The number of iterations hitting the maximum tree depth (0) is OK.
```

E-BFMI:

```
The E-BFMI (chain_1: 0.8652, chain_2: 0.9719, chain_3: 0.9089, chain_4: 0.9275) is OK.
```

General MCMC diagnostics

$\widehat{R}$:

```
All R-hat values are OK.
```

Bulk-ESS:

```
All bulk-ESS values are OK.
```

Tail-ESS:

```
All tail-ESS values are OK.
```

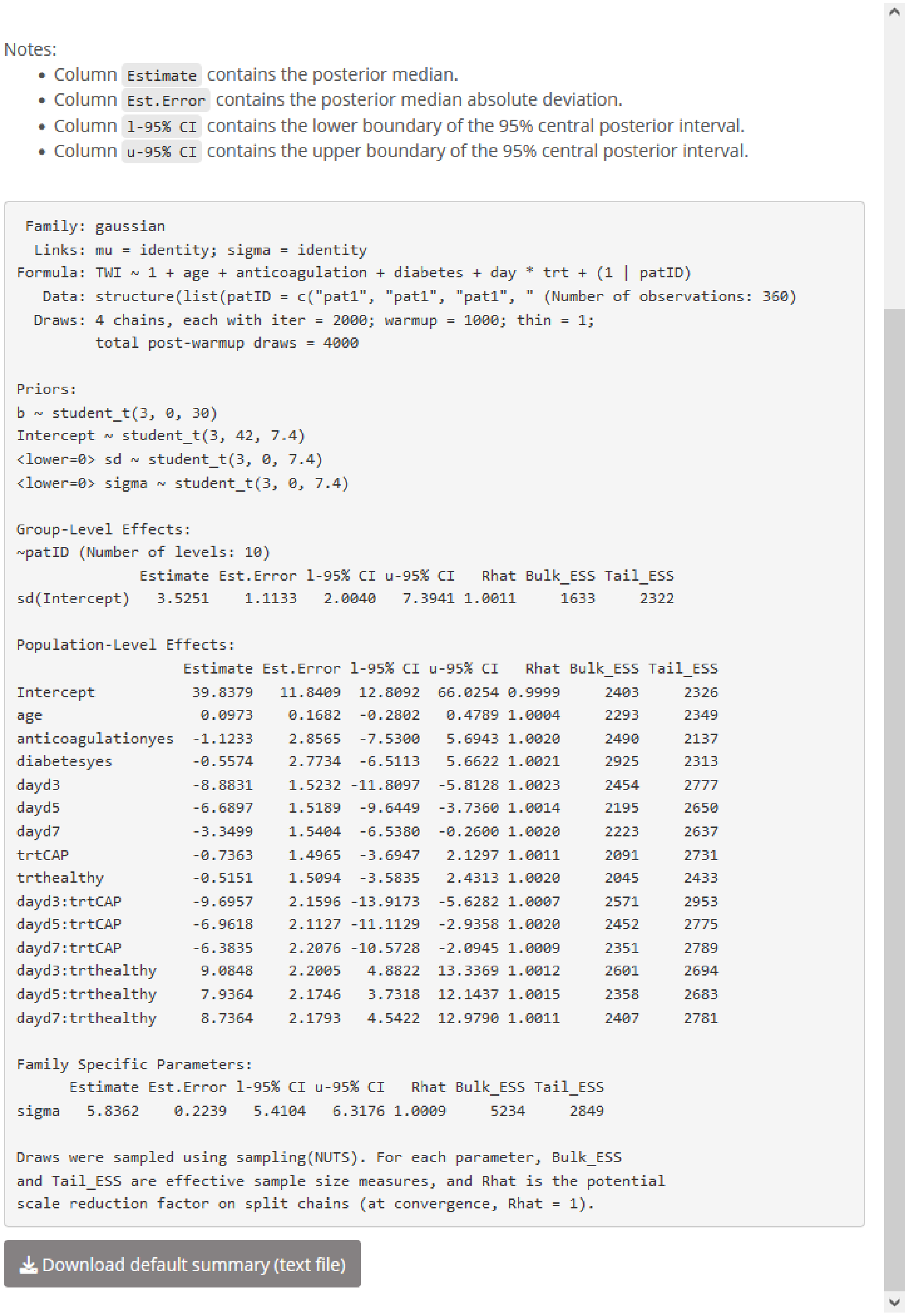
☐ Show detailed table of the general MCMC diagnostics

Figure 8: Tab "MCMC diagnostics" on page "Posterior". This tab presents the diagnostics from section Tab "MCMC diagnostics", applied to the user's Stan run (for the exact values of the general MCMC diagnostics, the checkbox "Show detailed table of the general MCMC diagnostics" needs to be checked). The purpose of this tab is to obtain details about problematic MCMC diagnostics in case there are such (after the Stan run, the user always receives a notification stating if there are problematic MCMC diagnostics or not).

7 Supplementary Material

This article comes with an online Supplement which consists of the following files:

- file 'Supplement_sections.pdf' which is a document with the following sections:
  - "Existing GUIs",
  - "Advanced distributional families",
  - "Frequentist analysis of the example";
- file 'CAP.csv' which contains the dataset for section Example;
- file 'weber_shinybrms.R' which contains the R code for section Example;
- file 'weber_shinybrms_sessionInfo.txt' which contains the original computing environment information for section Example. Note that the reproducibility of Stan results depends on the machine's hardware, so in general, our results from section Example will not be perfectly reproducible on other machines.

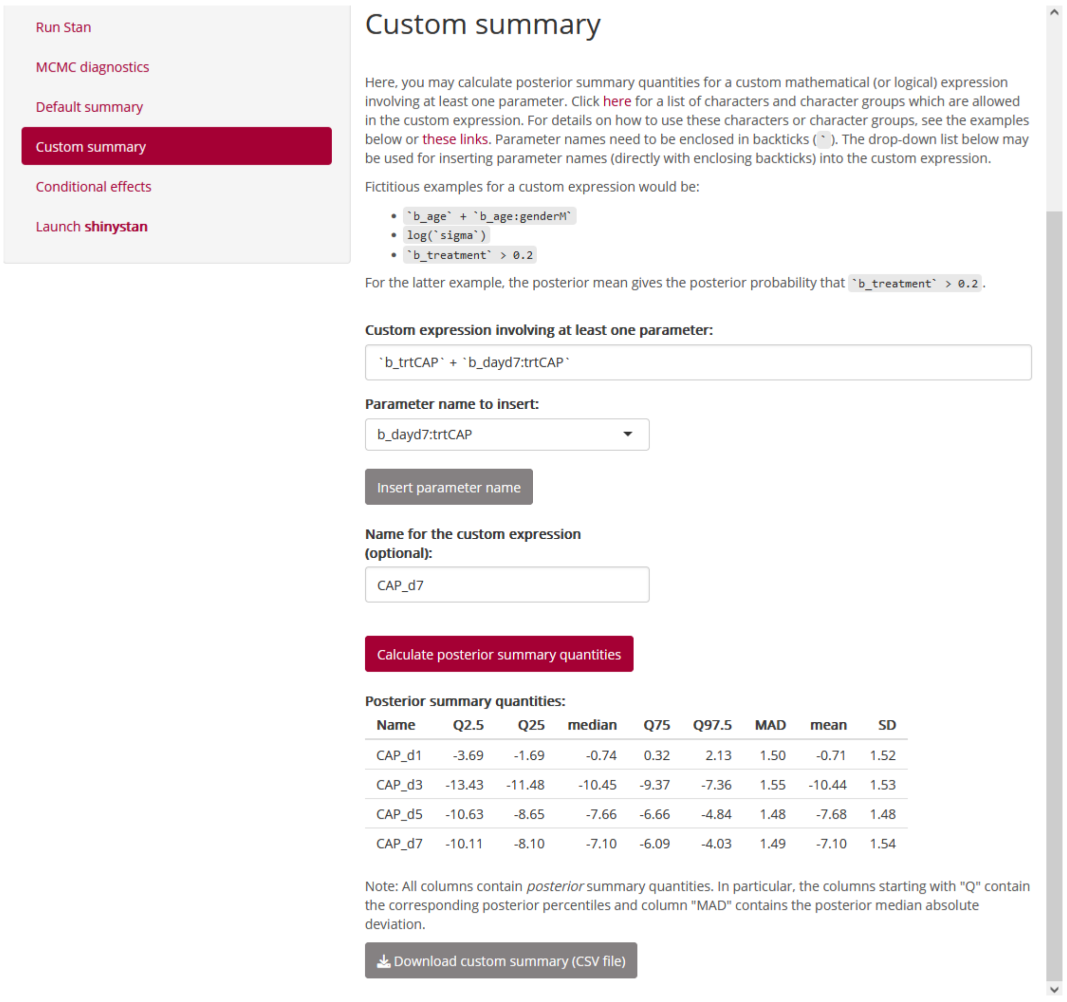


Notes:

- Column `Estimate` contains the posterior median.
- Column `Est.Error` contains the posterior median absolute deviation.
- Column `l-95% CI` contains the lower boundary of the 95% central posterior interval.
- Column `u-95% CI` contains the upper boundary of the 95% central posterior interval.

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: TWI ~ 1 + age + anticoagulation + diabetes + day * trt + (1 | patID)
   Data: structure(list(patID = c("pat1", "pat1", "pat1", " (Number of observations: 360)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Priors:
b ~ student_t(3, 0, 30)
Intercept ~ student_t(3, 42, 7.4)
<lower=0> sd ~ student_t(3, 0, 7.4)
<lower=0> sigma ~ student_t(3, 0, 7.4)

Group-Level Effects:
~patID (Number of levels: 10)
              Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
sd(Intercept)   3.5251    1.1133   2.0040   7.3941 1.0011     1633     2322

Population-Level Effects:
                   Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
Intercept           39.8379   11.8409  12.8092  66.0254 0.9999     2403     2326
age                  0.0973    0.1682  -0.2802   0.4789 1.0004     2293     2349
anticoagulationyes  -1.1233    2.8565  -7.5300   5.6943 1.0020     2490     2137
diabetesyes         -0.5574    2.7734  -6.5113   5.6622 1.0021     2925     2313
dayd3               -8.8831    1.5232 -11.8097  -5.8128 1.0023     2454     2777
dayd5               -6.6897    1.5189  -9.6449  -3.7360 1.0014     2195     2650
dayd7               -3.3499    1.5404  -6.5380  -0.2600 1.0020     2223     2637
trtCAP              -0.7363    1.4965  -3.6947   2.1297 1.0011     2091     2731
trthealthy          -0.5151    1.5094  -3.5835   2.4313 1.0020     2045     2433
dayd3:trtCAP        -9.6957    2.1596 -13.9173  -5.6282 1.0007     2571     2953
dayd5:trtCAP        -6.9618    2.1127 -11.1129  -2.9358 1.0020     2452     2775
dayd7:trtCAP        -6.3835    2.2076 -10.5728  -2.0945 1.0009     2351     2789
dayd3:trthealthy     9.0848    2.2005   4.8822  13.3369 1.0012     2601     2694
dayd5:trthealthy     7.9364    2.1746   3.7318  12.1437 1.0015     2358     2683
dayd7:trthealthy     8.7364    2.1793   4.5422  12.9790 1.0011     2407     2781

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI   Rhat Bulk_ESS Tail_ESS
sigma   5.8362    0.2239   5.4104   6.3176 1.0009     5234     2849

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

Figure 9: Tab "Default summary" on page "Posterior". Presented is the output of method `brms:::summary.brmsfit()` with arguments `priors` and `robust` set to `TRUE` (causing the priors to be shown, too, and the more robust summary quantities median and median absolute deviation to be used instead of the less robust quantities mean and standard deviation). This tab is only intended for a quick inspection (the **shinystan** app offers a more comprehensive output).



Custom summary

Here, you may calculate posterior summary quantities for a custom mathematical (or logical) expression involving at least one parameter. Click here for a list of characters and character groups which are allowed in the custom expression. For details on how to use these characters or character groups, see the examples below or these links. Parameter names need to be enclosed in backticks (`` ` ``). The drop-down list below may be used for inserting parameter names (directly with enclosing backticks) into the custom expression.

Fictitious examples for a custom expression would be:

- `` `b_age` + `b_age:genderM` ``
- `` log(`sigma`) ``
- `` `b_treatment` > 0.2 ``

For the latter example, the posterior mean gives the posterior probability that `` `b_treatment` > 0.2 ``.

Custom expression involving at least one parameter:

`b_trtCAP` + `b_dayd7:trtCAP`

Parameter name to insert:

b_dayd7:trtCAP

Name for the custom expression (optional):

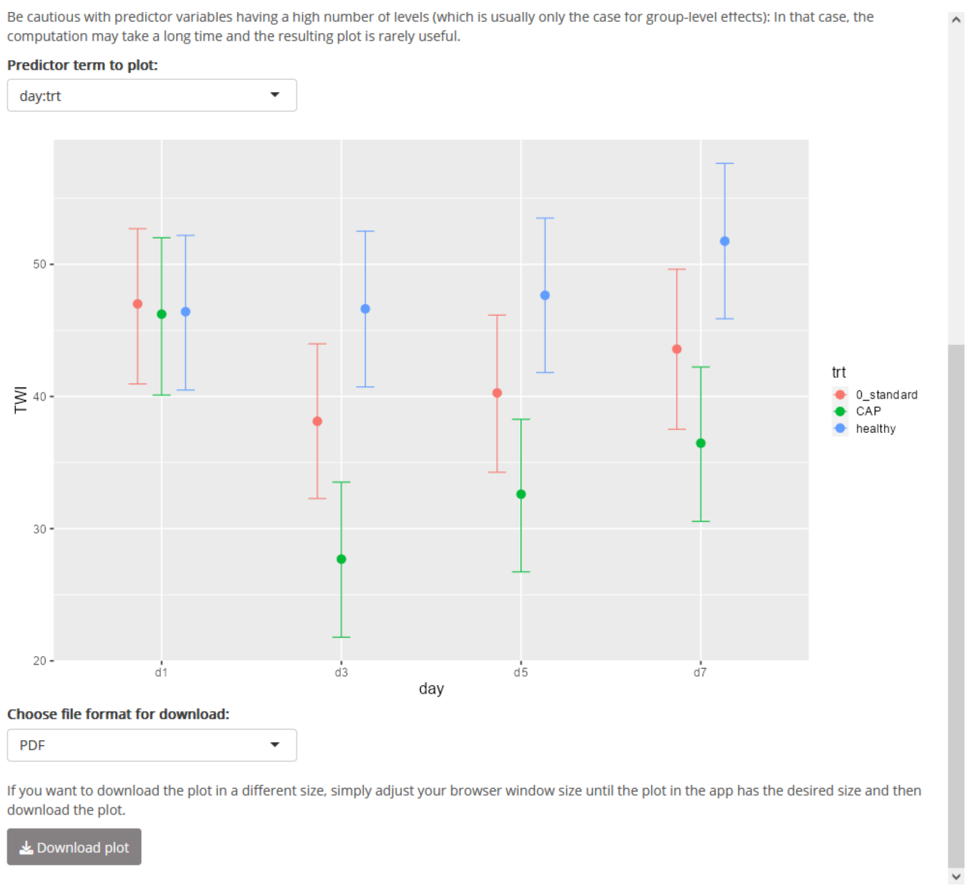
CAP_d7

Posterior summary quantities:

Name	Q2.5	Q25	median	Q75	Q97.5	MAD	mean	SD
CAP_d1	-3.69	-1.69	-0.74	0.32	2.13	1.50	-0.71	1.52
CAP_d3	-13.43	-11.48	-10.45	-9.37	-7.36	1.55	-10.44	1.53
CAP_d5	-10.63	-8.65	-7.66	-6.66	-4.84	1.48	-7.68	1.48
CAP_d7	-10.11	-8.10	-7.10	-6.09	-4.03	1.49	-7.10	1.54

Note: All columns contain *posterior* summary quantities. In particular, the columns starting with "Q" contain the corresponding posterior percentiles and column "MAD" contains the posterior median absolute deviation.

Figure 10: Tab "Custom summary" on page "Posterior". In contrast to tab "Default summary" (Figure 9), users can request their own summary quantities here. In the example presented here, we calculate the day-specific `CAP` effects. These show that apart from day 1 (where the wound areas had not been treated yet), the CAP treatment leads to a lower (i.e., better) TWI compared to the standard treatment (which is the reference category), with the posterior median ranging from ca. -10.45 to ca. -7.10 on days 3, 5, and 7.


Be cautious with predictor variables having a high number of levels (which is usually only the case for group-level effects): In that case, the computation may take a long time and the resulting plot is rarely useful.

Predictor term to plot:

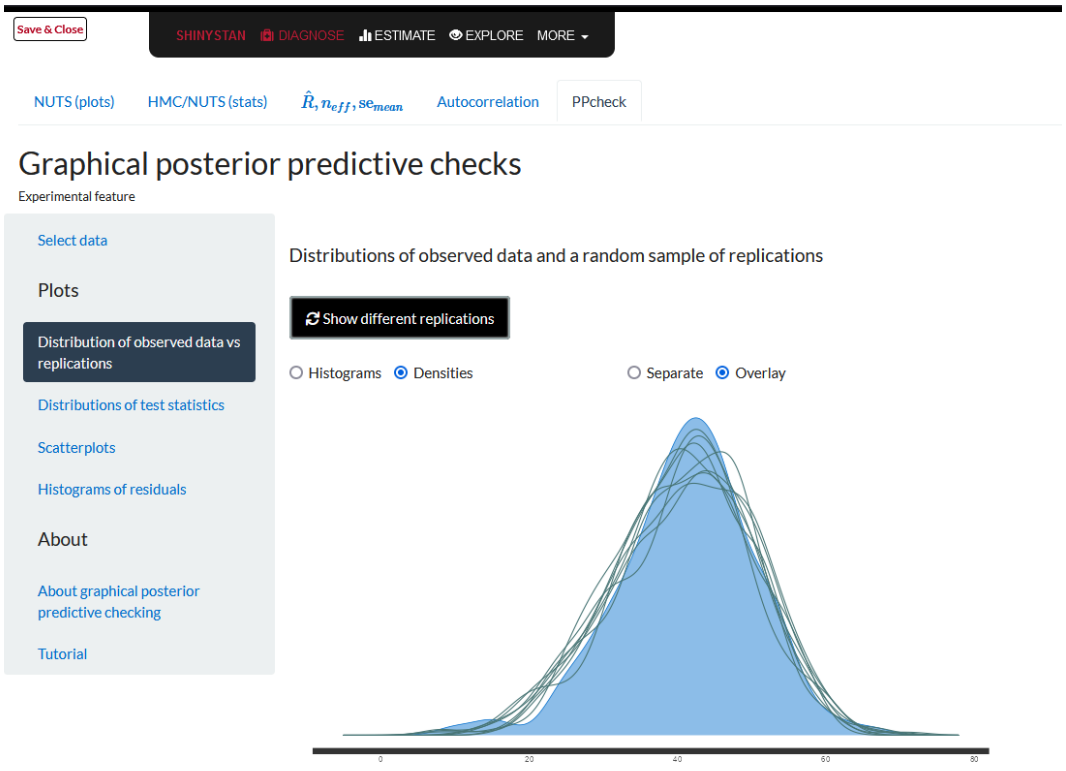
day:trt

Choose file format for download:

PDF

If you want to download the plot in a different size, simply adjust your browser window size until the plot in the app has the desired size and then download the plot.

Figure 11: Tab "Conditional effects" on page "Posterior". This tab shows the conditional-effects plots produced by `brms::conditional_effects()`. In the example presented here, we select the conditional-effects plot for the `day:trt` interaction. Similarly to Figure 10, this demonstrates that apart from day 1, the CAP treatment leads to a lower TWI compared to the standard treatment. This plot also illustrates that in the healthy skin area, the TWI is roughly constant over time, with a slight increase on day 7.


Save & Close
SHINYSTAN
DIAGNOSE
ESTIMATE
EXPLORE
MORE
NUTS (plots)
HMC/NUTS (stats)
$\hat{R}, n_{eff}, se_{mean}$
Autocorrelation
PPcheck
Select data
Plots
Distribution of observed data vs replications
Distributions of test statistics
Scatterplots
Histograms of residuals
About
About graphical posterior predictive checking
Tutorial
Show different replications
0
20
40
60
80

Graphical posterior predictive checks

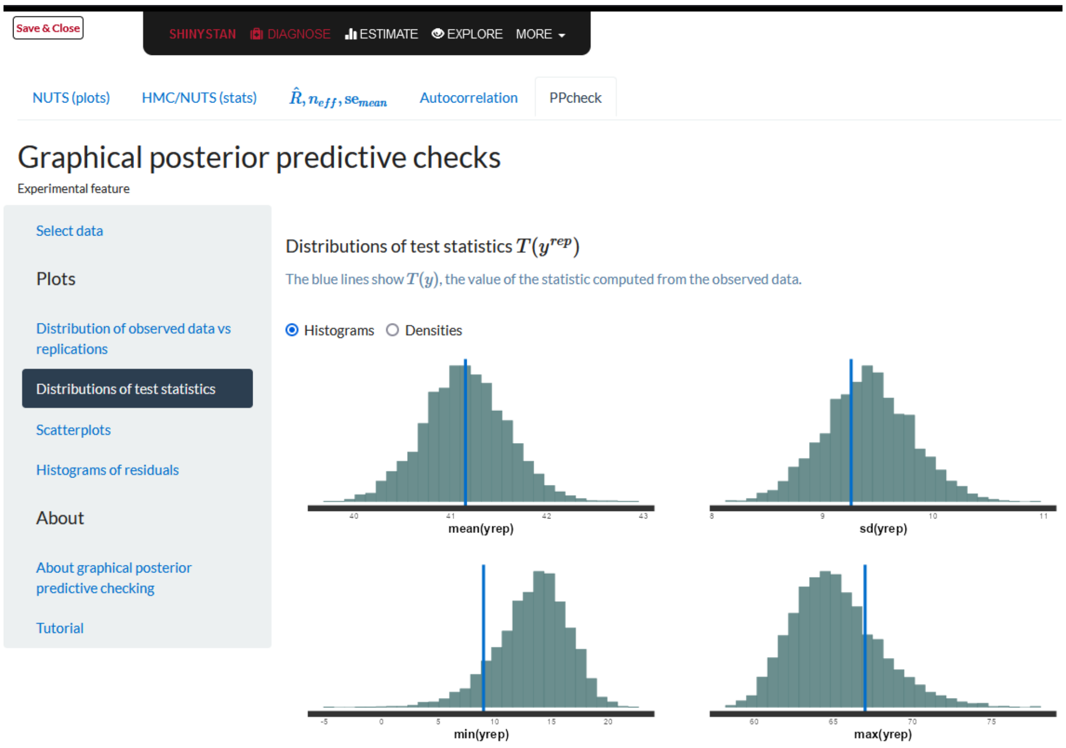
Experimental feature

Distributions of observed data and a random sample of replications

Histograms [✓ Densities]

Separate [✓ Overlay]

Figure 12: shinystan: PPC *via* overlaid kernel density estimates. The shaded blue density corresponds to the observed outcome values whereas each of the 8 overlaid green density lines corresponds to one randomly chosen post-warmup MCMC iteration. Here, the distributions of the model's predictions (which are based on the posterior, i.e., on the joint parameter distribution inferred from the data and the prior) are similar to the distribution of the observed outcome values, showing that at least in this regard, the model is a reasonable one.



Graphical posterior predictive checks

Experimental feature

Distributions of test statistics $T(y^{rep})$

The blue lines show $T(y)$, the value of the statistic computed from the observed data.

Figure 13: shinystan: PPCs *via* summary statistics. In contrast to the PPC from Figure 12, these PPCs here are based on *all* posterior draws which is possible by aggregating across the observations. The aggregation statistics are the mean (top left), the standard deviation (top right), the minimum (bottom left), and the maximum (bottom right). Here, these aggregated predictions show some room for model improvement: The minimum is overestimated—or rather "overpredicted"—by the model, the maximum is underestimated. Thus, the range of the replicated outcome values is narrower than the observed one. In contrast, the mean TWI is replicated reliably. The standard deviation shows a slight overestimation by the model. In summary, the Gaussian family seems to be a suboptimal outcome family, but we consider it to be sufficient for answering the primary research question (the comparison of CAP and standard treatment in terms of the central tendency of TWI values).

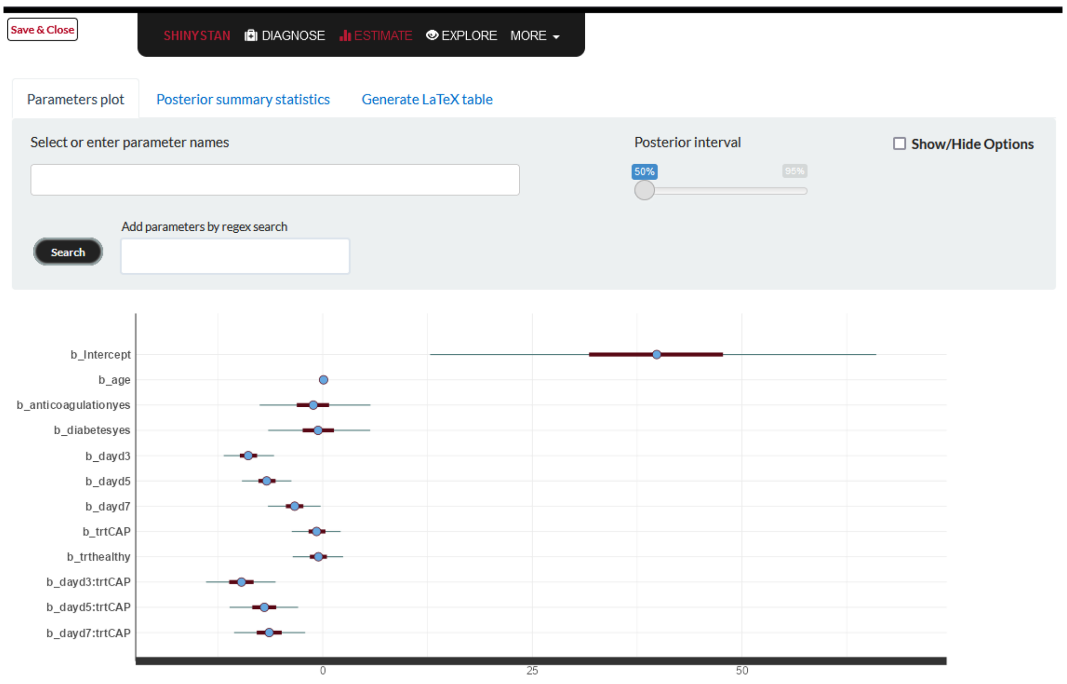


Figure 14: shinystan: Posterior intervals (credible intervals, CrIs). The default plot shown here is restricted to the first 12 parameters. More parameters may be selected in the two input fields above the plot. The different scales of the parameters (in particular, the intercept and the regression coefficient for age are on strikingly different scales) illustrate that in interactive use, it often makes sense to customize the selection of parameters.

Bibliography

J. Albert and J. Hu. *Probability and Bayesian Modeling*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, Boca Raton, FL, USA, 2019. ISBN 978-1-351-03014-4. URL https://doi.org/10.1201/9781351030144. [p96]

V. Amrhein and S. Greenland. Remove, rather than redefine, statistical significance. *Nature Human Behaviour*, 2(1):4, 2018. URL https://doi.org/10.1038/s41562-017-0224-0. [p97]

V. Amrhein, S. Greenland, and B. McShane. Scientists rise up against statistical significance. *Nature*, 567(7748):305, 2019. URL https://doi.org/10.1038/d41586-019-00857-9. [p97]

J. L. Arbuckle. *Amos*. IBM SPSS, Chicago, IL, USA, 2020. Version 27.0. [p99]

BEsmarter Team. *BEsmarter*, 2020a. URL https://besmarter-team.shinyapps.io/BEsmarter-GUI/. Web application (shiny app). Accessed on October 15, 2020. [p99]

BEsmarter Team. BEsmarter source code, 2020b. URL https://github.com/besmarter/BSTApp. Accessed on April 21, 2020. [p99]

M. Betancourt. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv:1701.02434v2 [stat]*, 2018. URL https://arxiv.org/abs/1701.02434v2. Accessed on March 7, 2021. [p98, 103]

P.-C. Bürkner. brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software*, 80(1):1–28, 2017. URL https://doi.org/10.18637/jss.v080.i01. [p97, 100, 101]

P.-C. Bürkner. Advanced Bayesian multilevel modeling with the R package brms. *The R Journal*, 10(1): 395–411, 2018. URL https://doi.org/10.32614/RJ-2018-017. [p97, 100, 101]

B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell. Stan: A probabilistic programming language. *Journal of Statistical Software*, 76 (1):1–32, 2017. URL https://doi.org/10.18637/jss.v076.i01. [p97]

W. Chang, J. Cheng, J. J. Allaire, C. Sievert, B. Schloerke, Y. Xie, J. Allen, J. McPherson, A. Dipert, and B. Borges. *shiny: Web Application Framework for R*, 2022. URL https://CRAN.R-project.org/package=shiny. R package, version 1.7.2. [p97]

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195 (2):216–222, 1987. URL https://doi.org/10.1016/0370-2693(87)91197-X. [p98]

G. Emvalomatis. *BayES: Bayesian Econometrics Software*, 2020. URL https://bayeconsoft.com/. Version 2.5. Accessed on November 24, 2020. [p99]

J. Gabry. *shinystan: Interactive Visual and Numerical Diagnostics and Posterior Analysis for Bayesian Models*, 2022. URL https://mc-stan.org/shinystan/. R package, version 2.6.0. [p97, 104]

J. Gabry, D. Simpson, A. Vehtari, M. Betancourt, and A. Gelman. Visualization in Bayesian workflow. *Journal of the Royal Statistical Society, Series A (Statistics in Society)*, 182(2):389–402, 2019. URL https://doi.org/10.1111/rssa.12378. [p98]

J. Gabry and R. Češnovar. *cmdstanr: R Interface to 'CmdStan'*, 2022. URL https://mc-stan.org/cmdstanr. R package, version 0.5.2. [p97]

A. E. Gelfand and A. F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990. URL https://doi.org/10.1080/01621459.1990.10476213. [p98]

A. Gelman. Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian Analysis*, 1(3):515–534, 2006. URL https://doi.org/10.1214/06-BA117A. [p96]

A. Gelman, A. Jakulin, M. G. Pittau, and Y.-S. Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, 2(4):1360–1383, 2008. URL https://doi.org/10.1214/08-AOAS191. [p96]

A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, Boca Raton, FL, USA, 3rd edition, 2014. ISBN 978-1-4398-4095-5. URL https://doi.org/10.1201/b16018. [p96, 98]

A. Gelman, D. Simpson, and M. Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555, 2017. URL https://doi.org/10.3390/e19100555. [p96]

A. Gelman, J. Hill, and A. Vehtari. *Regression and Other Stories*. Analytical Methods for Social Research. Cambridge University Press, Cambridge, UK, 2020a. ISBN 978-1-107-67651-0. URL https://doi.org/10.1017/9781139161879. [p96, 97]

A. Gelman, A. Vehtari, D. Simpson, C. C. Margossian, B. Carpenter, Y. Yao, L. Kennedy, J. Gabry, P.-C. Bürkner, and M. Modrák. Bayesian workflow. *arXiv:2011.01808v1 [stat]*, 2020b. URL https://arxiv.org/abs/2011.01808v1. Accessed on March 7, 2021. [p97]

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984. URL https://doi.org/10.1109/TPAMI.1984.4767596. [p98]

B. Goodrich, J. Gabry, I. Ali, and S. Brilleman. *rstanarm: Bayesian applied regression modeling via Stan*, 2022. URL https://mc-stan.org/rstanarm. R package, version 2.21.3. [p101]

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. URL https://doi.org/10.1093/biomet/57.1.97. [p98]

M. D. Hoffman and A. Gelman. The no-U-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(47):1593–1623, 2014. URL https://jmlr.org/papers/v15/hoffman14a.html. [p98]

IBM Corp. *IBM SPSS Statistics for Windows*. IBM Corp., Armonk, NY, USA, 2020. Version 27.0. [p99]

JASP Team. *JASP*, 2022. URL https://jasp-stats.org/. Version 0.16.3. [p99]

A. A. Johnson, M. Q. Ott, and M. Dogucu. *Bayes Rules!: An Introduction to Applied Bayesian Modeling*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, New York, NY, USA, 2022. ISBN 978-0-429-28834-0. URL https://doi.org/10.1201/9780429288340. [p96]

G. Karabatsos. A menu-driven software package for Bayesian regression analysis. *The ISBA Bulletin*, 22(4):13–16, 2015. [p96, 99]

G. Karabatsos. A menu-driven software package of Bayesian nonparametric (and parametric) mixed models for regression analysis and density estimation. *Behavior Research Methods*, 49(1):335–362, 2017. URL https://doi.org/10.3758/s13428-016-0711-7. [p99]

D. Lewandowski, D. Kurowicka, and H. Joe. Generating random correlation matrices based on vines and extended onion method. *Journal of Multivariate Analysis*, 100(9):1989–2001, 2009. URL https://doi.org/10.1016/j.jmva.2009.04.008. [p102]

D. Lunn, D. Spiegelhalter, A. Thomas, and N. Best. The BUGS project: Evolution, critique and future directions. *Statistics in Medicine*, 28(25):3049–3067, 2009. URL https://doi.org/10.1002/sim.3680. [p98]

D. J. Lunn, A. Thomas, N. Best, and D. Spiegelhalter. WinBUGS - A Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, 10(4):325–337, 2000. URL https://doi.org/10.1023/A:1008929526011. [p99]

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2003. [p98]

Y. Marchenko and N. Balov. In the spotlight: Bayesian "random-effects" models, 2015. URL https://www.stata.com/stata-news/news30-2/bayesian-random-effects/. Accessed on July 16, 2020. [p101]

R. McElreath. *Statistical Rethinking: A Bayesian Course with Examples in R and Stan*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, Boca Raton, FL, USA, 2nd edition, 2020. ISBN 978-0-367-13991-9. URL https://doi.org/10.1201/9780429029608. [p96]

B. B. McShane, D. Gal, A. Gelman, C. Robert, and J. L. Tackett. Abandon statistical significance. *The American Statistician*, 73:235–245, 2019. URL https://doi.org/10.1080/00031305.2018.1527253. [p97]

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. URL https://doi.org/10.1063/1.1699114. [p98]

R. M. Neal. Probabilistic Inference using Markov Chain Monte Carlo Methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993. URL https://www.cs.toronto.edu/~radford/review.abstract.html. Accessed on March 21, 2022. [p98]

R. M. Neal. MCMC using Hamiltonian dynamics. In *Handbook of Markov Chain Monte Carlo*, Handbooks of Modern Statistical Methods. CRC Press, Boca Raton, FL, USA, 2011. ISBN 978-0-429-13850-8. URL https://doi.org/10.1201/b10905. [p98]

J. Piironen, M. Paasiniemi, A. Catalina, F. Weber, and A. Vehtari. *projpred: Projection Predictive Feature Selection*, 2022. URL https://mc-stan.org/projpred/. R package, version 2.1.2. [p109]

H. Qian. *Toolkit on Econometrics and Economics Teaching*. MATLAB Central File Exchange, 2011. URL https://www.mathworks.com/matlabcentral/fileexchange/32601-toolkit-on-econometrics-and-economics-teaching. MATLAB package, version from August 19, 2011. Accessed on March 26, 2020. [p99]

A. Ramírez-Hassan and M. Graciano-Londoño. A GUIded tour of Bayesian regression. *The R Journal*, 13(2):135–152, 2021. URL https://doi.org/10.32614/RJ-2021-081. [p97, 99]

B. J. Reich and S. K. Ghosh. *Bayesian Statistical Methods*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, New York, NY, USA, 2019. ISBN 978-0-429-20229-2. URL https://doi.org/10.1201/9780429202292. [p96]

rollApp, Inc. rollApp - Run desktop applications online, 2020. URL https://www.rollapp.com/. Accessed on April 24, 2021. [p109]

rollApp, Inc. and JASP Team. JASP on rollApp, 2020. URL https://www.rollapp.com/app/jasp. Accessed on June 29, 2020. [p109]

D. Spiegelhalter, A. Thomas, N. Best, and D. Lunn. *OpenBUGS User Manual, Version 3.2.3*, 2014. URL http://www.openbugs.net/Manuals/Manual.html. Accessed on March 25, 2020. [p99]

Stan Development Team. *Stan Reference Manual, Version 2.29*, 2022a. URL https://mc-stan.org/docs/2_29/reference-manual/index.html. Accessed on April 13, 2022. [p98]

Stan Development Team. *RStan: The R Interface to Stan*, 2022b. URL https://mc-stan.org/. R package, version 2.21.5; for the example, version 2.26.13 from the repository at https://mc-stan.org/r-packages/ was used. [p97]

Stan Development Team. Runtime warnings and convergence problems, 2022c. URL https://mc-stan.org/misc/warnings.html. Version from March 10, 2022. Accessed on April 13, 2022. [p103]

Stan Development Team. *Stan Modeling Language Users Guide and Reference Manual, Version 2.29*, 2022d. URL https://mc-stan.org. [p97]

StataCorp. Introduction to Bayesian analysis. In *Stata Bayesian Analysis Reference Manual*. Stata Press, College Station, TX, USA, 2019a. ISBN 978-1-59718-272-0. URL https://www.stata.com/manuals/bayesintro.pdf. Release 16. Accessed on November 11, 2020. [p96]

StataCorp. *Stata*. StataCorp LLC, College Station, TX, USA, 2019b. Release 16. [p99]

A. Van Welzen, M. Hoch, P. Wahl, F. Weber, S. Rode, J. K. Tietze, L. Boeckmann, S. Emmert, and A. Thiem. The response and tolerability of a novel cold atmospheric plasma wound dressing for the healing of split skin graft donor sites: A controlled pilot study. *Skin Pharmacology and Physiology*, 34 (6):328–336, 2021. URL https://doi.org/10.1159/000517524. [p104]

A. Vehtari, A. Gelman, D. Simpson, B. Carpenter, and P.-C. Bürkner. Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC (with discussion). *Bayesian Analysis*, 16(2):667–718, 2021. URL https://doi.org/10.1214/20-BA1221. [p103]

F. Weber. *shinybrms: Graphical User Interface ('shiny' App) for 'brms'*, 2022. URL https://fweber144.github.io/shinybrms/. R package, version 1.8.0. [p97, 100]

F. Weber, G. Knapp, Ä. Glass, G. Kundt, and K. Ickstadt. Interval estimation of the overall treatment effect in random-effects meta-analyses: Recommendations from a simulation study comparing frequentist, Bayesian, and bootstrap methods. *Research Synthesis Methods*, 12(3):291–315, 2021. URL https://doi.org/10.1002/jrsm.1471. [p97]

P. Woodward. BugsXLA: Bayes for the common man. *Journal of Statistical Software*, 14(1):1–18, 2005. URL https://doi.org/10.18637/jss.v014.i05. [p97, 98]

P. Woodward. *Bayesian Analysis Made Simple: An Excel GUI for WinBUGS*. Chapman & Hall/CRC Biostatistics Series. CRC Press, Boca Raton, FL, USA, 2011. ISBN 978-1-4398-3954-6. URL https://doi.org/10.1201/b11235. [p99]

Frank Weber
Institute for Biostatistics and Informatics in Medicine and Ageing Research
Rostock University Medical Center
Ernst-Heydemann-Str. 8
18057 Rostock
Germany
ORCiD: 0000-0002-4842-7922
frank.weber@uni-rostock.de

Katja Ickstadt
Department of Statistics
TU Dortmund University
Vogelpothsweg 87
44227 Dortmund
Germany
ORCiD: 0000-0001-5157-2496
ickstadt@statistik.tu-dortmund.de

Änne Glass
Institute for Biostatistics and Informatics in Medicine and Ageing Research
Rostock University Medical Center
Ernst-Heydemann-Str. 8
18057 Rostock
Germany
ORCiD: 0000-0002-7715-9058
aenne.glass@uni-rostock.de